11 1994 ISSN 0542-9951

informatyka

P. 1877/94

informatyka

nr 11 1994
Listopad
Rok wydania XXX

Miesięcznik
ISSN 0542-9951
INDEKS 36124

KOLEGIUM REDAKCYJNE:

mgr Jarosław DEMINET
mgr inż. Piotr FUGLEWICZ
prof. dr hab. Jan GOLIŃSKI
mgr Teresa JABŁOŃSKA
(sekretarz redakcji)
mgr Krystyna
KARWICKA-RYCHLEWICZ
Władysław KLEPACZ
(redaktor naczelny)
mgr inż. Jan RYŻKO
dr Witold STANISZKIS
dr hab. Kazimierz SUBIETA
dr Zdzisław SZYJEWSKI

PRZEWODNICZĄCY
RADY PROGRAMOWEJ:

Prof. dr hab.
Juliusz Lech KULIKOWSKI

WYDAWCA:
Wydawnictwo Czasopism i Książek
Technicznych SIGMA NOT
Spółka z o.o.
ul. Ratuszowa 11
00-950 WARSZAWA
skrytka pocztowa 1004

Redakcja:
01-552 Warszawa,
Pl. Inwalidów 10, p. 104, 105
tel., fax: 39-14-34

Materiałów nie zamówionych
redakcja nie zwraca

W sprawach ogłoszeń prosimy zwracać się bezpośrednio do Redakcji lub Działu Reklamy i Marketingu 00-950 Warszawa ul. Mazowiecka 12 telefon: 27-43-66 telefaks: 26-80-16

W numerze:

	Strona
I Kongres Informatyki Polskiej	1
Szanse i zagrożenia – z Teresą Możdżyńską, wiceprezesem Głównego Urzędu Statystycznego rozmawiają *Krystyna Karwicka* i *Władysław Klepacz*	2
Zastosowanie sieci neuronowych w systemach czasu rzeczywistego – Halina Kwaśnicka, Urszula Markowska-Kaczmar	3
POSTGRES – rozszerzalny system zarządzania bazą danych – *Marcin Januchta, Zbyszko Królikowski*	7
Ankieta sieciowa – przyg. i oprac. *Krystyna Karwicka*	14
Systemy czasu rzeczywistego	
Synchronizacja w rozproszonych systemach uwarunkowanych czasowo – *Elżbieta Kosmulska-Bochenek*	17
Ze świata	
Największe firmy informatyczne w 1993 r. – *Jan Ryżko*	22
Osobiste stacje robocze	26

W najbliższych numerach:

- **Marcin Januchta i Zbyszko Królikowski omawiają system zarządzania POSTGRES jako narzędzie budowy aktywnych i dedukcyjnych systemów baz danych.**
- **Halina Kwaśnicka charakteryzuje zastosowanie algorytmów genetycznych w informatyce.**
- **Zbigniew Fryźlewicz i Jędrzej Ułasiewicz przedstawiają najistotniejsze własności systemu QNX 4 jako narzędzia tworzenia aplikacji czasu rzeczywistego.**

Warunki prenumeraty na 1995 r.

Zamówienia na prenumeratę czasopism wydawanych przez Wydawnictwo SIGMA-NOT można składać w dowolnym terminie. Mogą one obejmować dowolny okres czasu, tzn. dotyczyć dowolnej liczby kolejnych zeszytów każdego czasopisma.

Zamawiający może otrzymywać zaprenumerowany przez siebie tytuł począwszy od następnego miesiąca po dokonaniu wpłaty. Zamówienia na zeszyty sprzed daty otrzymania wpłaty będą realizowane w miarę możliwości – z posiadanych zapasów magazynowych.

Warunkiem przyjęcia i realizacji zamówienia jest otrzymanie z banku potwierdzenia dokonania wpłaty przez prenumeratora. Dokument wpłaty jest równoznaczny ze złożeniem zamówienia.

Wpłat na prenumeratę można dokonywać na ogólnie dostępnych blankietach w Urzędach Pocztowych (przekazy pieniężne) lub Bankach (polecenie przelewu), przekazując środki pod adres:
Wydawnictwo SIGMA-NOT Spółka z o.o.
Zakład Kolportażu
00-716 Warszawa, skr. poczt. 1004
konto:
PBK S.A. III 0/Warszawa nr 370015-1573-139-11

Wpłaty na prenumeratę przyjmują także wszystkie urzędy pocztowe nadawczo-odbiorcze oraz doręczyciele na terenie całego kraju

Na blankiecie wpłaty należy czytelnie podać nazwę zamawianego czasopisma, liczbę zamawianych egzemplarzy, okres prenumeraty oraz własny adres.
Na życzenie prenumeratora, zgłoszone np. telefonicznie, Zakład Kolportażu ul. Bartycka 20, 00-950 Warszawa, (telefony: 40-30-86, 40-35-89 oraz 40-00-21 wew. 249, 293, 299) wysyła specjalne blankiety zamówień wraz z aktualną listą tytułów i cennikiem czasopism.

Odbiorcy zagraniczni mogą otrzymywać czasopisma poprzez prenumeratę dewizową (wpłata dokonywana poza granicami Polski w dewizach, wg cennika dewizowego z cenami podanymi w dolarach amerykańskich) lub poprzez zamówioną w kraju prenumeratę ze zleceniem wysyłki za granicę (zamawiający podaje dokładny adres odbiorcy za granicą, dokonując równocześnie wpłaty w wysokości dwukrotnie wyższej niż cena normalnej prenumeraty krajowej).

Egzemplarze archiwalne (sprzedaż przelewowa lub za zaliczeniem pocztowym) można zamawiać pisemnie, kierując zamówienia pod adresem: Wydawnictwo SIGMA NOT, Spółka z o.o. Zakład Kolportażu, 00-716 Warszawa, ul. Bartycka 20, paw. B, tel. 40-37-31, natomiast za gotówkę można je nabyć w Klubie Prasy Technicznej w Warszawie ul. Mazowieckiej 12, tel. 26-80-17.

Istnieje możliwość zaprenumerowania 1 egz. czasopisma po cenie ulgowej przez indywidualnych członków stowarzyszeń naukowo-technicznych zrzeszonych w FSNT oraz przez uczniów zawodowych i studentów szkół wyższych. Blankiet wpłaty na prenumeratę ulgową musi być opatrzony na wszystkich odcinkach pieczęcią koła SNT lub szkoły.

W przypadku zmiany cen w okresie objętym prenumeratą Wydawnictwo zastrzega sobie prawo do wystąpienia o dopłatę różnicy cen oraz prawo do realizowania prenumeraty tylko w pełni opłaconej.
Cena jednego egzemplarza: normalna 45 000 zł, ulgowa 31 500 zł
Wartość prenumeraty w zł:
Normalna: kwartalna 135 000, półroczna 270 000, za trzy kwartały 404 000, roczna 540 000
Ulgowa: kwartalna 94 500, półroczna 189 000, za trzy kwartały 283 500, roczna 378 000

Skład i druk: Drukarnia SIGMA NOT Sp. z o.o. z. 388/94

P. 1877/94

POLITECHNIKA ŚLĄSKA Biblioteka Główna

I KONGRES INFORMATYKI POLSKIEJ

I Kongres Informatyki Polskiej organizowany w dniach 1–3 grudnia br. w Poznaniu stworzy nowe podejście do dyskusji, sposobu formułowania wniosków oraz określania programów działania na przyszłość w informatyce polskiej.

Najważniejsze prace merytoryczne Kongresu podzielono na pięć tematycznych bloków roboczych:

- Rządowe przedsięwzięcia informatyczne;
- Warunki rozwoju polskiego oprogramowania;
- Świadomość społeczna w odbiorze informatyki;
- Zaspokajanie potrzeb kadrowych informatyki;
- Rozwój technologii teleinformatycznych.

W dniu 3 grudnia nastąpi plenarne podsumowanie Kongresu na uroczystej sesji z udziałem zaproszonych Gości Honorowych. Pan Waldemar Pawlak – prezes Rady Ministrów RP – objął przewodnictwo Komitetu Honorowego Kongresu. Swój udział potwierdzili przedstawiciele wszystkich zaproszonych grup, tzn. Edukacji i szkoleń, Użytkowników oraz Producentów.

Jednym z istotnych elementów Kongresu będzie dyskusja i omówienie wyników specjalnie opracowanej ankiety dotyczącej informatyki w Polsce. O wypełnienie formularzy zostały poproszone wszystkie Autorytety związane z informatyką w Polsce. Poniżej publikujemy tę ankietę.

Ankieta dla uczestników I Kongresu Informatyki Polskiej

Uprzejmie prosimy o wypełnienie poniższej ankiety wymieniając w punktach najważniejsze elementy odpowiedzi **na wybrane bądź wszystkie pytania.**

Na podstawie uzyskanego zbioru elementów odpowiedzi zostanie przygotowany formularz zbiorczy, który w trakcie sesji środowiskowych na Kongresie zostanie poddany wartościowaniu, dając w rezultacie w miarę rzetelny obraz poglądów środowisk na przyszłość rozwoju informatyki w Polsce.

Zapraszamy do poświęcenia nieco swojego czasu na przemyślenie postawionych pytań i udzielenie na nie odpowiedzi. Będziemy również wdzięczni za wszystkie dodatkowe uwagi i propozycje. Mogą one być też włączone w wypowiedź pisemną przygotowaną do materiałów kongresowych.

A. Odbiór społeczny informatyki

1. Jak oceniasz obecny stan edukacji informatycznej społeczeństwa i w jaki sposób można go zmienić?
2. Co wpływa pozytywnie, a co negatywnie na wizerunek informatyki w społeczeństwie?
3. Kto i jak powinien kształtować wizerunek informatyki w społeczeństwie?
4. Co uznajesz za największe osiągnięcie w sferze informatyki w Polsce w ostatnim roku?
5. Co uznajesz za największą wpadkę informatyczną w Polsce w ostatnim roku?

B. Edukacja informatyczna

1. Jakie są podstawowe czynniki hamujące przygotowania nowych kadr informatycznych w Polsce?
2. Jakie są podstawowe potrzeby w zakresie edukacji informatycznej w szkolnictwie średnim?
3. Jakie są podstawowe czynniki gwarantujące rzetelne prowadzenie szkoleń informatycznych?
4. Jaka jest rola mediów w kształtowaniu obrazu rynku informatycznego?

C. Uwarunkowania prawne informatyki

1. Jakie kroki należy podjąć dla efektywnego wprowadzenia w życie przepisów ustawy o ochronie praw autorskich w zakresie oprogramowania?
2. Jakie czynniki mogą hamować proces legalizacji oprogramowania w Polsce?
3. Jakich jeszcze ustaw i decyzji prawnych brakuje w sferze informatyki?
4. Jakich zmian w polityce podatkowej należy dokonać dla zapewnienia rozwoju informatyki w Polsce?
5. Jakich zmian w polityce celnej należy dokonać dla zapewnienia rozwoju informatyki w Polsce?

D. Potrzeby zastosowań informatyki

1. Co twoim zdaniem może poprawić jakość zastosowań informatyki w Polsce?
2. Jakie są podstawowe czynniki gwarantujące rozwój zastosowań informatyki w Polsce?
3. Jakie produkty sprzętu komputerowego są obecnie najbardziej potrzebne dla rozwoju informatyki w Polsce?
4. Jakie produkty oprogramowania podstawowego są obecnie najbardziej potrzebne dla rozwoju informatyki w Polsce?
5. Jakiego rodzaju usługi są obecnie najbardziej potrzebne dla rozwoju informatyki w Polsce?

E. Przedsięwzięcia informatyczne

1. Które dziedziny życia gospodarczego i społecznego powinny być w pierwszym rzędzie zinformatyzowane przynosząc realne korzyści?
2. Jakie kroki należy podjąć dla efektywnego wprowadzenia w Polsce infostrad?
3. Jakie kroki należy podjąć dla efektywnego wykorzystania w Polsce systemów informacyjnych?
4. Jakie elementy powinny odgrywać główną rolę w podejmowaniu decyzji o wyborze oferenta w postępowaniu przetargowym na systemy informatyczne?
5. Jakie aplikacje są obecnie najbardziej potrzebne dla rozwoju informatyki w Polsce?

F. Rozwój polskiej informatyki

1. Jakie są podstawowe czynniki gwarantujące rozwój firm informatycznych w Polsce?
2. Jaką działalność firm informatycznych w Polsce należy szczególnie wspierać poprzez decyzje administracyjne?
3. Jakie polskie produkty informatyczne powinny się w pierwszym rzędzie pojawić na rynku?
4. Jakie polskie produkty informatyczne mogą stać się towarem eksportowym?
5. Jakie czynniki mogą znacznie zwiększyć obroty na rynku informatycznym dla użytkowników instytucjonalnych?
6. Jakie czynniki mogą znacznie zwiększyć obroty na rynku informatycznym powszechnego (domowego) użytku?

Uwaga: Odpowiedzi na wybrane pytania z ankiety oraz pisemną wypowiedź (do dwóch stron) prosimy przesłać na dyskietce w jednym z formatów AmiPro, WordPerfect, MS-Word lub ASCII (w ostateczności na papierze).

BIURO ORGANIZACYJNE
00-503 WARSZAWA 15, skr. 44
ul. Żurawia 4a, p. 200
tel. (02) 621 76 26, 693-59-46
komertel 39121125, fax (02) 693-59-49

Szanse i zagrożenia

Z Teresą Możdżyńską wiceprezesem Głównego Urzędu Statystycznego rozmawiają Krystyna Karwicka i Władysław Klepacz

– Główny Urząd Statystyczny jest „fabryką" informacji, czyli towaru, który w dzisiejszym świecie uchodzi za strategiczny. W jaki sposób GUS stara się zaspokoić rosnący popyt państwa i społeczeństwa na informację?

– Na całym świecie statystyka, w związku z rosnącą rolą informacji i wzrostem zainteresowań nią, przechodzi ogromną transformację. Zmienia się sposób organizacji badań, wprowadza nowe metody, nowe uregulowania prawne. Zmiany są też wymuszane przez technologię informatyczną. U nas proces ten jest spotęgowany przechodzeniem gospodarki z systemu centralistycznego na rynkowy, z totalitarnego na demokratyczny. Oznacza to, że ze statystyki państwowej, zorientowanej na kontrolę wykonania planu i innych poleceń rządu przez przedsiębiorstwa, instytucje i całe społeczeństwo, przechodzimy na statystykę publiczną. Jej obowiązkiem jest opracowywanie oczekiwanych, niezbędnych nie tylko administracji państwowej, ale również społeczeństwu, ludziom biznesu, instytutom naukowym itp. opisu zjawisk społeczno-gospodarczych. Z konieczności dokonania takiej modyfikacji również w legislacji, po 2 latach dyskusji narodził się nowy projekt ustawy o statystyce. Został on w tym roku przyjęty przez rząd i przesłany do Sejmu. Mamy nadzieję, że wejdzie pod obrady parlamentu najpóźniej w I kwartale 1995 r. W tym projekcie bardzo wyraźnie określona jest rola statystyki jako obiektywnego, niezależnego obserwatora i informatora dla wszystkich użytkowników, z gwarancją równoczesnego dostępu do informacji, co też jest szalenie ważne. Zasadę równoczesnego dostępu już stosujemy. Ważne i aktualne informacje statystyczne, opracowane i gotowe do emisji, przekazujemy za pośrednictwem mass mediów społeczeństwu równocześnie z przekazywaniem ich organom władzy centralnej. Musimy publikować, zgodnie z prawem, ok. 60 wskaźników. Ich ulokowanie w pakiet, który można uzyskać za pomocą modemu, jest ze wszech miar pożądane, jeśli statystyka ma być publiczna. Już obecnie zrobiliśmy mały kroczek w tym kierunku – Departament Informacji udostępnia część informacji modemowo. Niestety, serwis ten „wisi" na jednej linii telefonicznej.

– Zmiany w sposobie pracy statystyków są też wymuszane przez technologię informatyczną. W jakim stopniu widoczne jest to w GUS?

– Informatyka polepszyła komfort pracy statystyki. Od niej zależy możliwość gromadzenia, szybkość przetwarzania i udostępniania danych. W naszej statystyce jest jednak jeszcze wiele „białych plam" informatycznych. Ze względu na poziom techniki – nastąpiła u nas decentralizacja zbierania informacji, na zachodzie zaś centralizacja. Stan wojenny i ograniczenia COCOM sprawiły, że poza jednym używanym IBM 4381 oraz dwoma niezbyt wydajnymi maszynami ICL i pewną liczbą PC – nic nam w tamtym okresie nie przybyło. Większość przetwarzania danych statystycznych robiono na maszynach ODRA i rejestratorach MERA 9150. Nadal ten stary, moralnie zużyty sprzęt jest w statystyce wykorzystywany. Wpływ na polepszenie sytuacji miała podpisana w 1989 r. deklaracja o współpracy z Eurostat – Europejskim Biurem Statystycznym. Za otrzymane z funduszu PHARE na informatykę niespełna 1 mln ECU kupiliśmy 11 minikomputerów HP, co umożliwiło zastąpienie nimi w 11 ośrodkach maszyn ODRA. Obecnie, na szczęście, znów zaczyna się inwestować w informatykę w statystyce. Dostaliśmy z PHARE w tym roku 12 mln ECU. W ramach projektu POLSTAT będzie realizowane 5 podstawowych zadań. Jest to modernizacja jednostek terenowych i naszych dwóch centralnych ośrodków obliczeniowych – w Warszawie i w Radomiu – pracujących bezpośrednio na rzecz GUS, automatyzacja Centralnej Biblioteki Statystycznej, archiwizacja danych statystycznych oraz zbudowanie sieci, przede wszystkim lokalnych. Upowszechnienie informatyki pozwoli statystykom na bezpośrednie wykonywanie części operacji, co skróci proces tworzenia informacji.

– Czy to znaczy, że statystycy będą wykorzystywać narzędzia informatyczne bez pośrednictwa informatyków?

– W statystyce muszą być zatrudnieni informatycy, to oni przygotowują aplikacje, projektują systemy i niezbędne oprogramowanie, ale później, dzięki mikrokomputerom, statystyk sam może zweryfikować informacje, które zostały przekazane do urzędu statystycznego, wykonać analizę statystyczną i dodatkowe obliczenia, które uzna za niezbędne. Staramy się, aby nasze aplikacje, zgodne z przyjętymi wewnętrznymi standardami informatycznymi, przygotowywane przez naszych informatyków w 11 ośrodkach w kraju, były przyjazne dla użytkownika. Poziom wyposażenia w sprzęt informatyczny jest jeszcze niedostateczny, co nie sprzyja uniezależnieniu się statystyków od pomocy informatyków. W GUS większość PC-tów, to stare XT i AT. Ale systematycznie szkolimy pracowników z myślą o czasach, gdy każdy statystyk będzie miał dostęp do komputera i baz danych.

– Powiedziała Pani, że aplikacje są zgodne z przyjętymi standardami informatycznymi. Czy te standardy obowiązują również urzędy dostarczające statystyce informacji?

– Niestety, nie mamy wpływu na wprowadzanie standardów do innych urzędów. Jeszcze gorsze jest to, że cały proces automatyzacji zbierania i przekazywania informacji postępuje zbyt wolno. Szacujemy, że tylko 60–70% gmin ma zautomatyzowanych część swoich ewidencji. Sytuacja jest znacznie bardziej niebezpieczna, jeśli chodzi o skutki braku skoordynowanych działań na poziomie administracji państwowej.

– Mówiąc wprost, dziś nieomal cały informacyjny system naszego państwa opiera się na GUS. Rodzi się pytanie czy statystyka jest równoznaczna z informacją?

– Statystyka nie inspiruje do tego, aby być głównym dostarczycielem i administratorem informacji. Resorty zdają sobie sprawę z tego, iż muszą stworzyć własne systemy informacyjne, jak to jest w krajach rozwiniętych. Większość informacji niezbędnych do zarządzania państwem jest tam gromadzona

dokończenie na s. 16

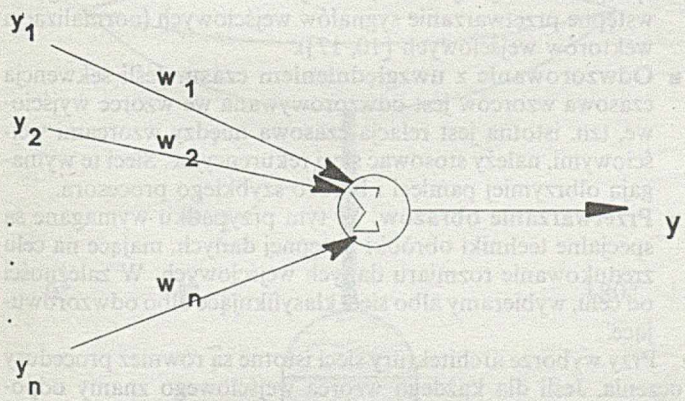
Zastosowanie sieci neuronowych w systemach czasu rzeczywistego

Halina Kwaśnicka
Urszula Markowska-Kaczmar
Centrum Informatyczne
Politechnika Wrocławska

Sieć neuronowa jest uproszczonym modelem mózgu. Składa się z pojedynczych elementów zwanych neuronami. Połączone ze sobą neurony tworzą sieć. Rys. 1 przedstawia pojedynczy neuron. Z każdym połączeniem związana jest modyfikowana waga w_i, która określa siłę połączenia i powoduje zmianę wpływu jednego neuronu na inny. Neuron zamienia przychodzące do niego sygnały wejściowe (wzorce wejściowe) w jeden sygnał wyjściowy, który przekazuje do innych neuronów. Konwersja ta wykonywana jest w dwóch etapach. Najpierw każdy przychodzący sygnał jest mnożony przez wagę na odpowiadającym mu połączeniu i tak ważone wejścia są sumowane, stanowiąc całkowity sygnał wejściowy do rozważanego neuronu. Opisuje to następująca równość:

$$x = \sum_i y_i w_i \qquad (1)$$

gdzie: y_i – i-ty sygnał wejściowy neuronu, w_i – waga na tym połączeniu.



Rys. 1. Pojedynczy neuron przekształca wzorce wejściowe we wzorzec wyjściowy

Otrzymany w ten sposób całkowity sygnał wejściowy x przekształcany jest przez neuron w sygnał wyjściowy y, przekazywany do innych neuronów. Przekształcenie to jest zależne od realizowanej przez neuron funkcji aktywacji (ang. *activation function*), określającej związek między całkowitym sygnałem wejściowym neuronu a jego odpowiedzią. Najczęściej jako funkcję aktywacji stosuje się funkcję: liniową, progową lub sigmoidalną.

Neurony w sieci mogą być połączone na różne sposoby. Struktura sieci może być warstwowa. Wyróżnia się warstwy: wejściową, wyjściową i jedną lub więcej warstw ukrytych. Aby sygnały pojawiające się na wyjściu były właściwe, czyli aby sieć właściwie realizowała stawiane jej zadanie, trzeba ją wyuczyć odpowiedzi na zadane wzorce wejściowe (proces trenowania sieci). Uczenie sieci polega na takim modyfikowaniu wag, aby sieć działała prawidłowo. Opracowano wiele algorytmów, które w procesie kolejnych iteracji pozwalają na wytrenowanie sieci.

Rodzaje sieci neuronowych

Ze względu na sposób przepływu sygnałów w sieci, funkcję aktywacji neuronów oraz zastosowany algorytm uczenia można wyróżnić wiele odmian sieci. Należy jednak podkreślić, że często stosuje się rozwiązania hybrydowe, łączące różne rodzaje sieci (patrz przykłady zastosowań sieci).

Obecnie najczęściej wykorzystywane są sieci warstwowe, w których stosuje się uczenie nadzorowane lub nienadzorowane. W systemach nadzorowanych określa się sytuację kiedy odpowiedź jest prawidłowa. Jeśli odpowiedź nie jest prawidłowa, sieć zaczyna korygować skojarzone z połączeniami wagi po to, aby zminimalizować błąd między otrzymanym sygnałem wyjściowym a wyjściem oczekiwanym. W nienadzorowanym uczeniu sieć jest systemem samoprzystosowującym się, który opierając się na wzorcach wejściowych potrafi sam określić wagi. Sieci takie potrafią grupować dane wejściowe i wyjściowe w określoną liczbę klas prototypów.

Jedną z najpopularniejszych metod nadzorowanego uczenia jest metoda propagacji wstecznej. Sieci, w których jest realizowana ta metoda uczenia, nazywane są sieciami z propagacją wsteczną. Przy zastosowaniu algorytmu propagacji wstecznej wykonywane są dwa przejścia. Pierwsze przejście do przodu, po to, aby wygenerować sygnał wyjściowy. W drugim przejściu, które jest w kierunku od warstwy wyjściowej do wejściowej, stosuje się procedurę korekcji błędów dla każdej warstwy tak, aby całkowity błąd między żądanym sygnałem wyjściowym i aktualnie otrzymanym sygnałem wyjściowym był minimalizowany. Aby zaimplementować tę procedurę należy zmieniać każdą wagę proporcjonalnie do stosunku wartości zmiany błędu i zmiany wagi. Proces ten jest wykonywany dla każdego wzorca sygnałów wejściowych sieci z zestawu uczącego [10, 17].

Inny ważny rodzaj sieci, to sieci ze sprzężeniem zwrotnym, nazywane inaczej rekurencyjnymi. Ogólnie jest to kategoria sieci, w których musi być więcej niż jeden obieg przez sieć zanim zostanie wyprodukowany sygnał wyjściowy. Przykładami takich sieci są: sieć Hopfielda, sieć BAM czy adaptacyjne sieci rezonansowe ART. Sieci rekurencyjne mogą być także konstruowane przy wykorzystaniu algorytmu wstecznej propagacji [8]. Sieci rekurencyjne z propagacją wsteczną są jednak ekstremalnie wolne, jeśli chodzi o proces uczenia.

Kolejnym rodzajem sieci są sieci wykorzystujące nienadzorowane uczenie z rywalizacją. Do takich należą między innymi: sieć SOM – *Self Organizing Map* (samoorganizujące się mapy), LVQ – *Learning Vector Quantization*, ELVQ – *Extended Learning Vector Quantization* i CNN – *Counterpropagation Neural Network*. W tego typu sieciach każdy sygnał wyjściowy

reprezentuje określoną klasę. Można je więc wykorzystywać do klasyfikacji wzorców wejściowych.

Rozpatrując zastosowanie sieci neuronowych do sterowania w czasie rzeczywistym, nie sposób nie wspomnieć o CMAC – *Cerebellum Model Articulation Controlers*. Mają wiele cech, które czynią je podobnymi do sieci z propagacją wsteczną, ale mają też niezaprzeczalną zaletę: mogą być uczone o wiele szybciej niż sieci z propagacją wsteczną. Jest to szczególnie istotne, jeśli sieć musi być trenowana w czasie rzeczywistym, np. w czasie kontroli ruchu powietrznego. Początkowo sieć CMAC była zaprojektowana dla robotów, jednakże ostatnio jest wykorzystywana do różnych zadań.

Należy również wspomnieć o sieciach stochastycznych, jak np. maszyna Boltzmana, która osiąga znacznie większą dokładność niż sieć z propagacją wsteczną.

Realizacje sieci neuronowych

Sieci neuronowe mogą być realizowane jako modele symulacyjne na uniwersalnych komputerach przy wykorzystaniu języków ogólnego przeznaczenia, np. C++. Tworzone są również specjalne pakiety oprogramowania dla potrzeb modelowania sieci, którego przegląd można znaleźć w [17] oraz [14]. Pozwalają one na zaprojektowanie, uczenie i modyfikacje sieci neuronowych. Adresowane są do różnych odbiorców, nawet do zupełnych laików w dziedzinie programowania, dla których opracowano pakiety komunikacji graficznej z użytkownikiem. Niektóre z nich są przeznaczone do projektowania sieci dla wybranej dziedziny, inne zaś do projektowania sieci w różnych dziedzinach zastosowań, jak np. system NeuroSmarts – do projektowania sieci wyspecjalizowanej w prognozowaniu rynku i finansów, w diagnozach medycznych, sterowania procesem, w rozpoznawaniu obrazów czy w predykcji i analizie uszkodzeń. Systemy te różnią się także liczbą możliwych do zrealizowania za ich pomocą architektur sieci, no i ceną. W obu wymienionych wyżej źródłach można również znaleźć listę sprzętu do zastosowań sieciowych. Sprzętu, w sensie specjalizowanych komputerów, jak również układów scalonych realizowanych w różnych technikach.

Prawie wszystkie firmy półprzewodnikowe czynią próby w kierunku produkcji odpowiednich układów scalonych, choć na razie dostępnych jest tylko kilka z nich. Implementacje sieci neuronowych w postaci sprzętowej wykorzystują różne techniki. Na plan pierwszy wysuwają się następujące techniki: analogowa, optoelektryczna, cyfrowa.

Typowa implementacja analogowa sieci neuronowych opiera się na pamięci stałej, jak EEPROM, aby pamiętać wagi sieci. Stale poszukuje się nowych możliwości wykonania układu zawierającego przestrajalne wagi (a więc zdolne do uczenia). W układach tych proces uczenia jest realizowany w sterującym komputerze, a następnie wartości te są wprowadzane do układu.

W układach optoelektrycznych działanie modelowanej sieci jest odwzorowywane przez przestrzenny modulator światła (SLM – *Spatial Light Modulator*), przetwarzający sygnały świetlne emitowane z matrycy świetlnej w n sygnałów świetlnych wyjściowych, podawanych na matrycę fotodekoderów. Zaletą układów optoelektrycznych jest szybkość ich działania, jednakże istnieją duże trudności przy próbie budowy takich układów dla sieci neuronowych o zmiennych wagach.

Układy z całkowicie cyfrowymi sieciami neuronowymi są kolejną alternatywą. Projekty zdążają ku temu, by upodobnić je do innych układów w komputerze. Prostota projektu cyfrowego pozwala na przeprowadzenie uczenia w czasie rzeczywistym. Mogą być one również połączone w kaskadę dla otrzymania większej sieci.

W wielu zastosowaniach pracujących w czasie rzeczywistym wykorzystuje się realizacje sprzętowe sieci neuronowych. Na ogół jednak realizacje sprzętowe sieci neuronowych nie mają modyfikowalnych wag, a więc nie można ich uczyć. Aby wyuczyć sieć określonego zadania, tzn. określić jej wagi, należy daną sieć zasymulować. Otrzymane w wyniku symulacji wagi są następnie wprowadzane do układu scalonego.

Wybór architektury i realizacji sieci do konkretnego zastosowania

Wybór architektury sieci i jej realizacji do konkretnego zastosowania jest zagadnieniem złożonym [7]. Po pierwsze należy rozpatrzyć, jaką informację możemy podać jako sygnały wejściowe do sieci oraz czego oczekujemy jako informacji wyjściowej sieci. Dalej określamy rodzaj zastosowania, nad którym pracujemy, przy czym w tym przypadku wyróżnia się następujące kategorie:

- **Odwzorowanie.** Zachodzi wtedy, gdy przekształcamy określone wzorce wejściowe w odpowiednie wzorce wyjściowe, np. gdy sygnały odbierane z czujnika są przekształcane w ustawienia zaworu bezpieczeństwa. W takich przypadkach najlepiej jest używać sieci z propagacją wsteczną (sieci SOM lub CNN).
- **Pamięć skojarzeniowa.** Jest to zapamiętywanie informacji przez kojarzenie jej z inną informacją. Jeśli sieć ma produkować jeden z wzorców wyjściowych, jakich ją wyuczyliśmy, z możliwością ponownego jego wywołania (różne wzorce wejściowe dają tę samą odpowiedź, jeśli są wystarczająco podobne), to mamy do czynienia z pamięcią skojarzeniową, a więc możemy zastosować np. sieć Hopfielda.
- **Klasyfikacja.** Zachodzi wtedy, gdy sygnały wejściowe muszą być grupowane w klasy i sygnałem wyjściowym jest klasa, do jakiej należy sygnał wejściowy. W tym przypadku wykorzystujemy sieci samoorganizujące się, np. sieć Kohonena, ART (Adaline, sieci stochastyczne). Realizacja tych sieci wymaga sprzętu o dużej szybkości, należy zwrócić również uwagę na wstępne przetwarzanie sygnałów wejściowych (normalizacja wektorów wejściowych [10, 17]).
- **Odwzorowanie z uwzględnieniem czasu.** Jeśli sekwencja czasowa wzorców jest odwzorowywana we wzorce wyjściowe, tzn. istotna jest relacja czasowa między wzorcami wejściowymi, należy stosować sieci rekurencyjne. Sieci te wymagają olbrzymiej pamięci i bardzo szybkiego procesora.
- **Przetwarzanie obrazów.** W tym przypadku wymagane są specjalne techniki obróbki wstępnej danych, mające na celu zredukowanie rozmiaru danych wejściowych. W zależności od celu, wybieramy albo sieci klasyfikujące albo odwzorowujące.

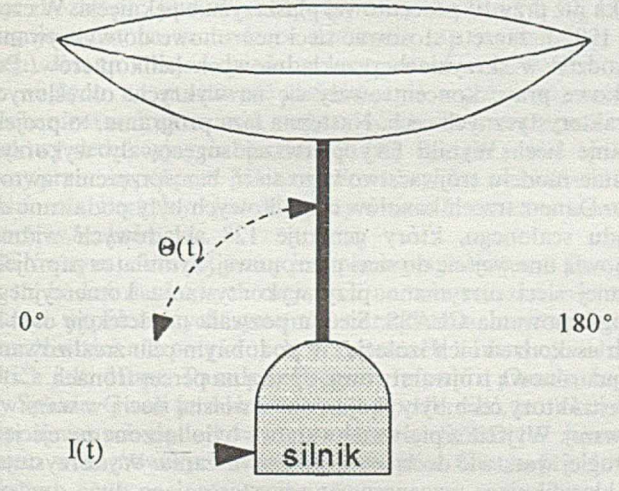
Przy wyborze architektury sieci istotne są również procedury uczenia. Jeśli dla każdego wzorca wejściowego znamy odpowiadające mu wyjście, możemy użyć uczenia nadzorowanego. Jeśli nie, to należy zastanowić się, czy możemy dostarczyć dokładnych informacji o pożądanym funkcjonowaniu sieci. Przy twierdzącej odpowiedzi na ostatnie pytanie wykorzystujemy uczenie klasyfikujące (ang. *graded*), tzn. dostarczamy miary stwierdzającej, jak dobry lub zły jest sygnał wyjściowy w stosunku do oczekiwanego. Jeśli w ogóle nie mamy takich informacji, wykorzystujemy procedury nienadzorowanego uczenia, a więc sieć Kohonena lub ART.

W przypadku systemów pracujących w czasie rzeczywistym istnieją również specjalne ograniczenia uzależnione np. od tego, czy sieć musi się uczyć na bieżąco (na bieżąco modyfikować wagi), czy też musi osiągać bardzo duże szybkości w czasie uczenia i przetwarzania oraz czy powinna uczyć się przez cały czas. Przy takich ograniczeniach sieć z propagacją wsteczną wybieramy tylko wówczas, gdy możemy zastosować specjalną kartę przyspieszającą lub jeśli pracujemy z komercyjnym układem scalonym.

Mimo tak wielu ograniczeń, jakie należy uwzględnić próbując wykorzystać sieci neuronowe do rozwiązania danego problemu, można jednak powiedzieć, że przeglądając znane z literatury rozwiązania, trud taki bardzo się opłaca.

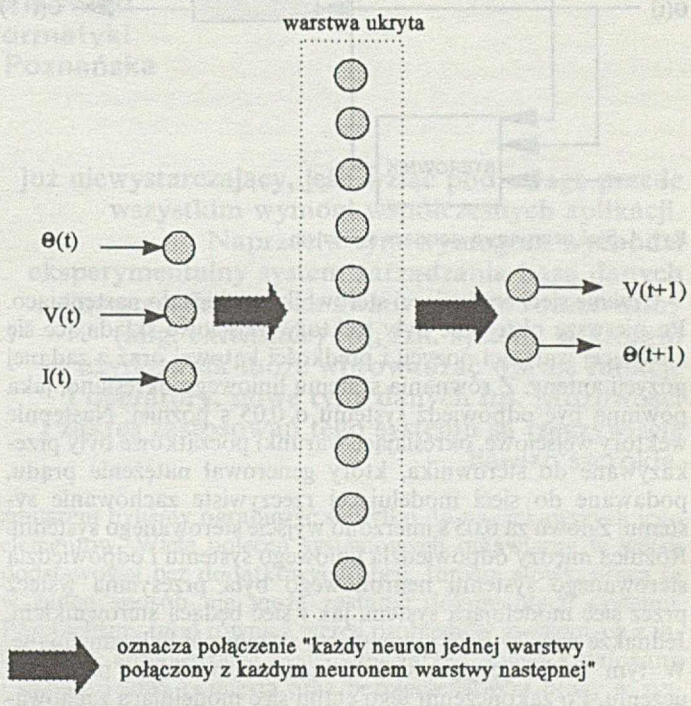
Przykłady zastosowań sieci neuronowych w systemach czasu rzeczywistego

Artykuł [1] prezentuje zastosowanie sieci neuronowych jako sterowników w systemach o charakterystyce nieliniowej. W czasie projektowania systemu sterującego sieć neuronowa była wykorzystywana dwukrotnie. Po raz pierwszy, do otrzymania matematycznego modelu rzeczywistego systemu, który miał podlegać sterowaniu. Po raz drugi, w celu zaprojektowania systemu sterującego rzeczywistym systemem. W obu przypadkach zastosowano sieć z propagacją wsteczną. Jako przykład systemu o charakterystyce nieliniowej posłużyła zamontowana na pręcie antena poruszana przez silnik (rys. 2). Moment obrotowy anteny jest proporcjonalny do wejściowego natężenia prądu *I*. Siła ciężkości i tarcie powodują zakłócenia ruchu anteny. Do znalezienia matematycznego modelu dynamiki rzeczywistego systemu zastosowano dwuwarstwową sieć z propagacją wsteczną, w której neurony w warstwie ukrytej realizowały sigmoidalną funkcję aktywacji neuronu, natomiast neurony w warstwie wyjściowej miały tę funkcję liniową. Sieć, służąca jako model systemu jest pokazana na rys. 3. Ma ona trzy neurony w warstwie wejściowej reprezentujące kolejno: pozycję kątową anteny θ, prędkość kątową v i wejściowe natężenie prądu *I*. Warstwa wyjściowa ma dwa neurony reprezentujące odpowiednio: pozycję kątową anteny z dokładnością do 1° oraz prędkość kątową anteny, określające stan systemu w następnej chwili, oznaczanej $(t + dt)$. Liczbę neuronów w warstwie ukrytej wybrano arbitralnie równą 10.



Rys. 2. Antena poruszana przez silnik

Uzyskanie danych do wytrenowania sieci (wzorców uczących) nie było trudne. Co 0,05 sekundy mierzono pozycję i prędkość kątową anteny. Wybrano 700 pomiarów. Każdy wektor wejściowy repezentował takie parametry anteny, jak: kąt, prędkość i natężenie prądu. Różnica między wyjściem sieci i wartościami systemu pomiarowego zainstalowanego na antenie służyły do obliczenia błędu podczas uczenia sieci. Po zastosowaniu wektora wejściowego jako bodźca do pierwszej warstwy, jest on przesyłany przez wszystkie wyższe warstwy aż do wygenerowania wyjścia. Wyjście to jest następnie porównywane z oczekiwanym wyjściem, w tym przypadku z wartością otrzymaną z systemu pomiarowego i jest obliczany błąd sygnału dla jednostki wyjściowej. Sygnał błędu jest następnie transmitowany z warstwy wyjściowej do warstwy pośredniej, ale każda jednostka w warstwie pośredniej otrzymuje tylko część sygnału całkowitego błędu, która odpowiada jej udziałowi w tworzeniu sygnału wyjściowego. Na podstawie otrzymanego sygnału błędu uaktualniane są wagi połączeń każdej jednostki. Powoduje to, że sieć dąży do stanu, w którym zakodowane są wszystkie wzorce uczące. Sieć jest uczona tak długo, aż suma kwadratów błędów osiągnie akceptowalną wartość. W ten sposób otrzymywany jest model rzeczywistego systemu. Sieć była budowana przy użyciu pakietu oprogramowania Neural Network Toolbox.



Rys. 3. Sieć neuronowa modelująca zachowanie anteny poruszanej przez silnik

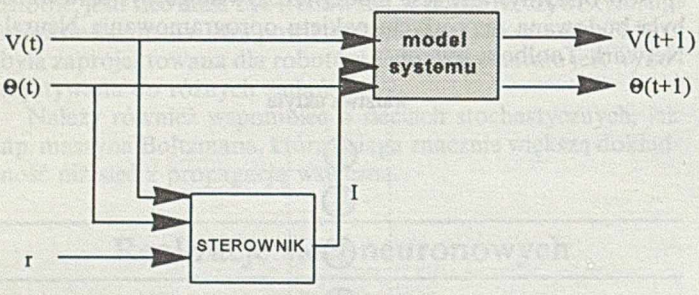
Przy okazji można się zastanawiać, jak to się dzieje, że sieć może dawać właściwą odpowiedź nawet na wektory wejściowe, które nie należały do zestawu wzorców uczących. Wielu badaczy wykazuje [10], że wszystkie jednostki w warstwach ukrytych są w jakiś sposób skojarzone ze specyficznymi cechami wzorców wejściowych jako wynik uczenia. Jakiego rodzaju skojarzenia tam występują, nie jest to na ogół jawne. Istotne jest, że sieć znajduje wewnętrzną reprezentację, która pozwala na generowanie oczekiwanego wyjścia na dany wzorzec wejściowy. Ta sama reprezentacja jest wykorzystywana do wygenerowania wyjścia na wektor wejściowy, który nie należał do zbioru trenującego.

Po uzyskaniu modelu rzeczywistego systemu, kolejnym krokiem było utworzenie nowej sieci neuronowej o podobnej strukturze, która po jej wytrenowaniu działa w systemie jako sterownik anteny. Model systemu, podlegającego sterowaniu i sterownik pokazano na rys. 4. Sieć realizująca sterownik ma strukturę podobną do pokazanej na rys. 3. W tym przypadku, w warstwie wejściowej występują trzy neurony, gdyż na wejście sterownika podawane są: pozycja kątowa anteny θ, prędkość kątowa v i zadane położenie anteny r. W warstwie wyjściowej występuje tylko jeden neuron, określający natężenie prądu; w warstwie ukrytej przyjęto, jak poprzednio, 10 neuronów. Zadaniem sterownika jest wygenerowanie takiego natężenia prądu *I*, aby antena znalazła się w stanie równowagi z dokładnością do 1° w stosunku do zadanego położenia, określonego wartością r. Założono, że sterowana sieć neuronowa będzie działać jak system o charakterystyce liniowej, tzn. założono, że

oczekiwane zachowanie sterowanego systemu będzie opisywane równaniem:

$$\frac{dV}{dt} = 9 * r - * \theta(t) - 6V(t) \qquad (2)$$

gdzie: V – prędkość kątowa anteny,
θ – pozycja kątowa anteny,
r – zadane położenie anteny.



Rys. 4. Sieć neuronowa sterująca systemem

Uczenie sieci realizującej sterownik przebiegało następująco. Po pierwsze określane były wektory wejściowe składające się z bieżącej wartości pozycji i prędkości kątowej oraz z zadanej pozycji anteny. Z równania systemu liniowego określano, jaka powinna być odpowiedź systemu o 0,05 s później. Następnie wektory wejściowe, określające warunki początkowe były przekazywane do sterownika, który generował natężenie prądu, podawane do sieci modelującej rzeczywiste zachowanie systemu. Znowu za 0,05 s mierzono wyjście sterowanego systemu. Różnica między odpowiedzią liniowego systemu i odpowiedzią sterowanego systemu neuronowego była przesyłana wstecz przez sieć modelującą system, jak i sieć będącą sterownikiem. Jednakże wagi w sieci modelującej system nie były zmieniane. W tym kroku jedynie sieć będąca sterownikiem podlegała uczeniu. Po zakończeniu tego etapu sieć modelująca zachowanie systemu antenowego nie jest już dłużej potrzebna. Przed użyciem sieciowego sterownika był on testowany przez porównanie odpowiedzi sterowanej anteny i systemu liniowego. Aby sprawdzić dokładność sieciowego sterownika, zadawano zmiany pozycji systemu liniowego i systemu neuronowego w zakresie od 10° do 90°. Okazało się, że sieciowy sterownik zachowywał się jak system liniowy, a więc może być wykorzystywany w rzeczywistym systemie.

W [9] opisano interesujący przykład zastosowania sieci neuronowej do sterowania samochodem. Jest to projekt wykonany w Carnegie Melon University. Dużą ciężarówkę z zainstalowanymi wieloma komputerami w środku wyposażono w kamerę wideo. Obraz pobierany z kamery jest podzielony na 960 piksli. Stanowi to wejście do sieci neuronowej, która jest prostą trzywarstwową siecią z propagacją wsteczną. Warstwa wejściowa zawiera 960 neuronów, warstwa środkowa tylko 5 neuronów. Warstwa wyjściowa składa się z 30 neuronów, z których każdy reprezentuje odpowiedni kierunek kątowy ruchu ciężarówki. Środkowe neurony warstwy wyjściowej reprezentują kierunek prosto do przodu, dalsze reprezentują odchylenia od tego kierunku. A więc wyjście z sieci było bezpośrednio wykorzystywane do sterowania kierunkiem jazdy ciężarówki. W przypadku jazdy samochodem, trudno jest wygenerować wszystkie możliwe scenariusze, jakie mogą zaistnieć w czasie jazdy.

Zastanawiano się więc, jak wytrenować sieć. Postanowiono wykorzystać człowieka jako nauczyciela, tzn. sieć otrzymywała sygnał wejściowy z kamery, wypracowywała odpowiedź wyjściową, która była porównywana z działaniem człowieka w takiej sytuacji. Różnica między nimi była błędem warstwy wyjściowej, następnie wykorzystywano propagację wsteczną do modyfikacji wag. Zauważono jednak, że nie było to wystarczające, aby móc rozważyć wszystkie możliwości, jakie mogą wystąpić w rzeczywistym świecie, np. jeśli na prostej drodze pojawiała się niewielka zmiana w obrazie z kamery, mogło to wpływać na sposób objechania rogu skrzyżowania. Aby tego uniknąć, utrzymywano bufor z ostatnimi 200 obrazami. Ponadto 15 obrazów przesuwano trochę w lewo i trochę w prawo, aby mieć większą perspektywę. Obrazów tych używano do ponownego trenowania sieci. Rezultaty, jakie osiągnięto, są wielce obiecujące. Ciężarówka poruszała się po drogach i ścieżkach dookoła Campusa, występowały tam zarówno zaśmiecone drogi i ścieżki dla rowerów, jak i dwupasmowa droga szybkiego ruchu. Ciężarówka osiągnęła szybkość 20 mil na godz. i mogła przejechać ponad połowę mili na nieznanym odcinku drogi.

Wiele problemów z dziedziny wojskowej próbuje się rozwiązać przy wykorzystaniu sieci neuronowych. W [16] opisano program monitorujący, jaki zainicjowały Siły Powietrzne USA pod nazwą *Health and Usage Monitoring* (HUM). Nowe operacje wymagają, aby systemy diagnostyczne na pokładzie były zdolne do dostarczenia obsłudze powietrznej informacji w czasie rzeczywistym, i to obojętne, czy odbywa się to w dżungli, czy na pustyni. Wybrano sieci neuronowe, gdyż oferują one znaczące zredukowanie złożoności przy zwiększonej niezawodności. Umożliwiają one diagnostykę na pokładzie w czasie rzeczywistym przez zbieranie danych z układu kierowniczego, układu napędowego, z systemu wirnikowego i olejowego. Szczególnie dobre rezultaty osiągnięto w diagnostyce silników helikopterów, a dokładniej w wykrywaniu uszkodzeń zmęczeniowych przekładni napędowych. Większość tego typu uszkodzeń musi być usuwana natychmiast, gdyż inaczej grozi to katastrofą.

W ostatnim dziesięcioleciu prowadzono wiele interesujących badań nad analizą sygnałów wibracyjnych w czasie rzeczywistym, w celu wykrycia początkowej fazy uszkodzeń zmęczeniowych. Testy laboratoryjne Boeinga wykazały, że taka metoda może wykryć znaczące zmiany w transmisji dźwięku pracy silnika już przy 10-procentowej płaszczyźnie pęknięcia. W czerwcu 1992 r. zaczęto stosować sieci neuronowe do wykrywania uszkodzeń w skrzyniach przekładniowych helikopterów. Początkowe prace koncentrowały się na wykryciu określonych charakterystycznych cech. Następna faza programu, to projektowanie sieci. Wyniki fazy pierwszej sugerowały wykorzystywanie modelu trójwarstwowego sieci, bez sprzężenia zwrotnego. Dane z trzech kanałów czujnikowych były podawane do układu scalonego, który generuje 128 składowych widma. Stanowią one wejście do sieci neuronowej. Symulator zaprojektowanej sieci otrzymano przy wykorzystaniu komercyjnego oprogramowania CLASS. Sieć ta pozwala na detekcję osobliwych uszkodzeń i ich izolację. W podobnym celu zrealizowano sieć neuronową trójwarstwową, opartą na perceptronach. Cztery ekstraktory cech były skojarzone z własną siecią w warstwie pierwszej. Wyjście z pierwszej warstwy było łączone z wejściem w drugiej warstwie do łącznego przetwarzania. Wykorzystując sieć klasyfikującą wygenerowano wystarczająco dużo danych, aby już przy dwudziestu obrotach wałka zróżnicować stan transmisji. W celu określenia cech dla sieci klasyfikującej, zastosowano specjalne urządzenie, które pozwoliło na redukcję cech dla sieci klasyfikującej do 10–15. Zbiór uczący miał 1000 elementów, zbiór testujący 6700 elementów. Prawdopodobieństwo fałszywego alarmu lub błędnej detekcji było bliskie zeru.

Nowe podejście w implementacji sieci neuronowych do sterowania procesem produkcyjnym zostało opisane w [15]. W artykule tym sieć ta jest oznaczana skrótem PCNN (ang. *Process Control Neural Network*) i powstała przez zastąpienie warstwy wyjściowej sieci stochastycznej, która jest implementacją klasyfikatorów wzorców Bayes'a przez warstwę sieci Grossberga (sieci ART). Warstwa wejściowa jest z kolei poprzedzona siecią Kohonena.

dokończenie na s. 27

POSTGRES
rozszerzalny system zarządzania bazą danych

Marcin Januchta
Zbyszko Królikowski
Instytut Informatyki
Politechnika Poznańska

Model danych wykorzystywany w tradycyjnych systemach zarządzania bazami danych (SZBD) składa się z zestawu relacji. Każdej relacji odpowiada schemat, który jest zbiorem atrybutów określonego typu. Możliwe typy atrybutów dostępne we współczesnych komercyjnych SZBD to: liczby całkowite, zmienno-przecinkowe, łańcuchy znaków, data i waluta. Niewątpliwą zaletą takiego modelu danych jest jego prostota. Jednakże na obecnym etapie rozwoju systemów baz danych model ten jest już niewystarczający, jeśli wziąć pod uwagę przede wszystkim wymogi współczesnych aplikacji. Naprzeciw tym wymogom wychodzi eksperymentalny system zarządzania bazą danych Postgres. Jest to tak zwany system rozszerzalny (ang. *extensible*) [6], tzn. system, w którym użytkownik może wprowadzać własne funkcje i operatory, nowe typy danych itp. Możliwości i zakres zastosowań tego systemu są rzeczywiście bardzo szerokie.

System może służyć m.in. do przetwarzania złożonych obiektów [7, 8], takich jak np. obrazy. Większość współczesnych systemów baz danych to systemy pasywne, wszystkie operacje manipulowania danymi są związane tylko i wyłącznie z realizacją poleceń użytkowników. Wyposażenie systemu zarządzania bazą danych w mechanizmy tworzenia i przetwarzania reguł daje możliwość budowy aktywnej bazy danych [4, 8], która reaguje sama na zachodzące zdarzenia (np. modyfikację danych). Jest to kolejna dziedzina zastosowań systemu Postgres. Ze względu na możliwość definiowania reguł i odpowiedni mechanizm wnioskowania, może on również stanowić podstawę budowy dedukcyjnych baz danych [8]. System został opracowany przez zespół prof. Michaela Stonebrakera na University of California, Berkeley i jest dostępny w sieci *Internet* na serwerze *epoch.cs.berkeley.edu*.

Głównym celem tej pracy jest prezentacja systemu Postgres na tle badań naukowych mających na celu rozszerzenie możliwości współczesnych systemów zarządzania bazami danych – możliwości niespotykanych w systemach konwencjonalnych takich jak: dBase, Paradox, Progres, Informix, Oracle, Ingres czy Sybase. Gwoli ścisłości należałoby podkreślić, że niektóre ze wspomnianych dodatkowych możliwości SZBD już wprowadzono w najnowszych wersjach systemów Oracle [1], Ingres i Sybase.

Niniejsza prezentacja systemu Postgres składa się z dwóch części. W części pierwszej omówimy: model danych systemu, język zapytań Postquel i możliwości jego rozszerzania przez użytkowników oraz bogatą bibliotekę LIBPQ służącą do komunikacji z systemem Postgres z poziomu programów w języku C. W drugiej części scharakteryzujemy możliwości budowy aktywnych i dedukcyjnych baz danych z wykorzystaniem tego systemu oraz omówimy manipulowanie obiektami złożonymi. Przedstawimy również prowadzone w Instytucie Informatyki Politechniki Poznańskiej prace nad rozszerzeniem tego systemu.

Chcielibyśmy zwrócić uwagę Czytelników, że ze względu na badzo szeroki zakres tego systemu, w niniejszej prezentacji zmuszeni byliśmy ograniczyć się tylko do najciekawszych jego elementów. Wiele poleceń systemowych omawiamy w wersji podstawowej bez dodatkowych opcji. Innymi słowy, praca ta w żaden sposób nie może zastąpić dokumentacji systemu. Pominęliśmy tutaj również szereg problemów związanych z instalacją i administracją systemem, zarządzaniem katalogami i współdzieloną pamięcią oraz bezpieczeństwem pracy w systemie.

* Praca częściowo finansowana w ramach grantu 43-0329 (KBN2).

Model danych systemu POSTGRES

Model danych systemu Postgres, w porównaniu z modelem tradycyjnym, oferuje kilka dodatkowych konstrukcji [5, 6], powiększających w znaczący sposób siłę wyrazu tego modelu. Są to: rozszerzalne typy atrybutów, klasy (ang. *classes*), dziedziczenie (ang. *inheritance*) i rozszerzalne funkcje. Omówimy obecnie te nowe, dodatkowe elementy systemu Postgres.

Tak jak w tradycyjnym modelu relacyjnym, baza danych systemu Postgres jest kolekcją relacji; każdej relacji odpowiada schemat, który jest zbiorem atrybutów określonego typu oraz kluczy. Jednakże w systemie Postgres nie używa się pojęcia relacji, lecz znacznie szerszego pojęcia **klasy.**

Jest to fundamentalne pojęcie tego systemu [2, 3]. **Klasa** jest zbiorem **wystąpień** (inaczej: instancji) obiektów, posiadającym określoną nazwę. Formalnie przez **obiekt** rozumie się dwójkę: **o = (oid, v)**, gdzie **oid** jest unikalnym identyfikatorem obiektu, a **v** jest wartością obiektu. Wartość obiektu, w zależności od jego struktury, może być wartością atrybutu, krotką wartości lub zbiorem wartości.

Struktura obiektu jest określona przez zbiór nazwanych atrybutów. **Atrybuty** przyjmują wartości z odpowiednich dziedzin, które są zbiorami wartości elementarnych, takich jak liczby naturalne, rzeczywiste czy łańcuchy znaków. Każde wystąpienie obiektu ma taki sam zbiór atrybutów określonego typu.

W jaki sposób użytkownik może utworzyć klasę oraz przykłady klas omówimy w następnym punkcie, który jest poświęcony językowi zapytań Postquel. Przedstawimy w nim również pozostałe elementy modelu danych systemu Postgres tj. **dzie-**

dziczenie i funkcje, bowiem pojęcia te najłatwiej można omówić posługując się konkretnymi przykładami z języka zapytań systemu.

Istotną różnicą, w porównaniu z modelem tradycyjnym, jest natomiast to, że w modelu danych systemu Postgres wyróżnia się dwa typy atrybutów [2, 3, 4]: **atomowe** (ang. *atomic*) i **strukturalne** (ang. *structured*). Typy atrybutów strukturalnych są definiowane przez użytkowników, natomiast typy atomowe są to przede wszystkim typy atrybutów zdefiniowane w systemie (tak zwane typy predefiniowane), ale mogą nimi również być typy atrybutów definiowane przez użytkowników. Predefiniowane typy atomowe to na przykład: *int*2, *int*4, *float*4, *bool*, *char*10, *date*, gdzie, np. *int*2 to typ liczb całkowitych, których binarna reprezentacja ma długość dwóch bajtów. Ze względu na możliwość wprowadzania przez użytkowników rozszerzeń w typach danych, w systemie Postgres mówimy o **abstrakcyjnych typach danych** [7] (ang. *abstract data types (ADT)*). W ogólności składają się one z definicji struktury oraz definicji operatorów, które mogą być stosowane wyłącznie dla tej struktury.

W systemie Postgres definicja struktury składa się z:

- nazwy typu,
- długości wewnętrznej reprezentacji w bajtach,
- procedur konwersji wartości z reprezentacji zewnętrznej na wewnętrzną i vice versa,
- wartości domyślnych.

Na przykład, typ *int*4 jest wewnętrznie definiowany następująco [2]:

```
define type int4
    (internallength = 4,
    input=int4in,
    output= int4out,
    default = "0" )
```

Procedury są dołączane do systemu przez wydanie polecenia **define function**. Operatory abstrakcyjnych typów danych są definiowane z wykorzystaniem polecenia **define operator**, przez podanie m.in.:

- liczby i typów operatorów,
- procedury stanowiącej implementację operatora,
- typu wyniku,
- prawa pierwszeństwa operatorów (ang. *precedence*).

Prawo pierwszeństwa operatorów jest parametrem opcjonalnym i jest określane przez liczbę całkowitą, przy czym im większa liczba, tym wyższy priorytet.

Na przykład, operator + dla typu *int*4 jest definiowany następująco:

```
define operator +
    (arg1=int4, arg2=int4,
     procedure = int4pl,
     precedence = 5,
     associativity = left)
```

Jako procedury obliczającej wynik działania operatora możemy użyć funkcji uprzednio zdefiniowanej poleceniem **define function** lub jednej z funkcji systemowych. **Int4pl** oznacza systemową procedurę implementującą dodawanie dwu czterobajtowych liczb całkowitych.

Niektóre inne możliwe operatory dla typu *int*4 to, na przykład [2]:

operator	typ	realizowany przez funkcję systemową
!=	bool	int4ne(int4,int4)
>	int4	int4gt(int4,int4)
/	int4	int4div(int4,int4)
/	int4	int42div(int4,int2)
*	int4	int4mul(int4,int4)

Operator można usunąć poleceniem **remove operator.** Ponieważ operator o tej samej nazwie może być różnie realizowany dla rozmaitych typów danych, należy tu podać, dla której kombinacji typów danych operator jest usuwany.

Prześledźmy teraz funkcjonowanie omówionych poleceń na konkretnym przykładzie. Przedstawiony niżej przykład jest skróconym zapisem sesji w systemie Postgres, w trakcie której zmienimy definicję operatora „>". Operator ten standardowo jest obliczany przez funkcję **int4gt**. Po zmianie jego definicji będzie on realizowany przez funkcję **intcmp**, zdefiniowaną wcześniej przez użytkownika poleceniem **define function.** Funkcja ta zamiast „x > y" oblicza:

„x > y + 100".

Załóżmy, że w bazie danych została zdefiniowana klasa *twoints*, zawierająca dwa atrybuty *x* i *y* typu *int*4 (dwie czteroznakowe liczby całkowite). Listę wystąpień tej klasy można uzyskać wydając polecenie **retrieve** – jego składnię i opcje omówimy w następnym punkcie – tutaj wystarczy nam jego intuicyjna interpretacja. Po wydaniu tego polecenia, system potwierdza jego przyjęcie wyświetlając komunikat: *„Query sent to backend is..."*.

```
* retrieve (twoints.all) where twoints.x>twoints.y \g
Query sent to backend is "retrieve (twoints.all) where twoints.x>twoints.y "
-------------------------
| x        | y        |
-------------------------
| 250      | 100      |
-------------------------
```

Usuniemy obecnie operator > wydając odpowiednie polecenie **remove.**

```
* remove operator > (int4,int4) \g
Query sent to backend is "remove operator > (int4,int4) "
REMOVE
Go
```

System potwierdził jego przyjęcie i oczekuje na wprowadzenie kolejnego polecenia. Wydanie polecenia **retrieve** z żądaniem wyświetlenia wystąpień klasy *twoints*, dla których powinien być spełniony warunek $x > y$, powoduje wyświetlenie sekwencji komunikatów systemowych przedstawionych poniżej, informujących użytkownika, że niestety nie ma operatora dla typu *int*4 (został usunięty wcześniej poleceniem **remove**) i zapytanie wydane w tej postaci nie może zostać wykonane. Użytkownik powinien sformułować zapytanie inaczej lub zdefiniować odpowiedni operator wykorzystując w tym celu polecenie **define operator.**

```
* retrieve (twoints.all) where twoints.x>twoints.y \g
Query sent to backend is "retrieve (twoints.all) where twoints.x>twoints.y "
NOTICE:Jan 13 18:28:27:there is no operator > for types int4 and int4
NOTICE:Jan 13 18:28:27:You will either have to retype this query using an
NOTICE:Jan 13 18:28:27:explicit cast, or you will have to define the operator
WARN:Jan 13 18:28:27:> for int4 and int4 using DEFINE OPERATOR
Go
```

Poniżej przedstawiono nową definicję operatora >, implementowaną z wykorzystaniem procedury **intemp,** napisanej przez użytkownika.

```
* define operator > is
  (arg1=int4, arg2=int4, procedure=intcmp) \g
Query sent to backend is "define operator > (arg1=int4, arg2=int4,
procedure=intcmp) "
DEFINE
Go
```

Wydanie obecnie polecenia **retrieve,** z żądaniem wyświetlenia wystąpień klasy *twoints,* dla których powinien być spełniony warunek $x > y$, spowoduje wyświetlenie tylko tych wartości atrybutów x i y, dla których zachodzi zależność $x > y + 100$ (bo tak został zdefiniowany operator > w procedurze **intemp**).

```
* retrieve (twoints.all) \g
Query sent to backend is "retrieve (twoints.all) "
-----------------------
| x       | y       |
-----------------------
| 100     | 90      |
-----------------------
| 250     | 100     |
-----------------------
Go
```

Wykorzystując metody omówione powyżej, użytkownik może do zbioru typów predefiniowanych w systemie, dodawać nowe, własne typy atomowe [5]. Załóżmy na przykład, że użytkownik pragnie zdefiniować nowy typ *„box"* – tutaj rozumiany jako prostokąt. Ma on być reprezentowany przez lewy górny i prawy dolny narożnik, to znaczy przez dwie liczby, które są współrzędnymi tych narożników:

```
define type box
    (internallength = 16,
    input= CharToBox,
    output = BoxToChar,
    default = " ")
```

W celu określenia czy dwa obrazy mają taki sam obszar można zdefiniować odpowiedni operator, na przykład =

```
define operator =
    (arg1=box, arg2=box,
    procedure = equal_area,
    precedence = 3,
    associativity = left,
    sort = BoxArea).
```

Podstawowym elementem definicji tego operatora jest procedura **equal_area**. Wprowadzono tutaj dodatkową własność **sort** definiowanego operatora, określającą procedurę, która powinna być wykorzystana dla posortowania klasy zawierającej wartości typu *box,* jeśli jest to konieczne podczas wykonywania zapytania.

O ile w przypadku predefiniowanych typów danych takich jak *int*4 konwersja do i z reprezentacji wewnętrznej jest dokonywana automatycznie, to w przypadku typów definiowanych przez użytkownika niezbędne jest jawne wskazanie typu, do którego należy tej konwersji dokonać. Na przykład sekwencja

$''(1, 1, 4, 5)''$

jest interpretowana jako napis. Aby zasygnalizować systemowi, że jest to zewnętrzna reprezentacja obiektu typu *box* o wierzchołkach (1, 1) i (4, 5), należy napisać to w następującej postaci

$''(1, 1, 4, 5)'' :: \text{box}$

Wykorzystując polecenie **create**, możemy teraz zdefiniować klasę *obraz* z atrybutem *a* typu *box.*

```
create obraz(a = box)
append obraz(a = "(1, 1, 4, 5)"::box)
```

Polecenie **append** służy do wprowadzania nowych wystąpień do klasy – dla naszego przykładu do klasy *obraz.*

Strukturalne typy danych reprezentując obiekty złożone mogą być definiowane z wykorzystaniem tak zwanych, konstruktorów macierzy (ang. *array constructor*) i procedur [2, 3, 6]. Obiektem złożonym może być właśnie macierz o stałym lub zmiennym rozmiarze. Dla przykładu

```
char [25]
```

definiuje macierz znakową o stałym rozmiarze 25 znaków. Jeśli górna granica zostanie pominięta, to uzyskujemy definicję macierzy o zmiennym rozmiarze. Analogicznie można zdefiniować tablicę składającą się z 10 elementów typu *int*4 jako:

```
int4 [10]
```

Można również utworzyć nowy typ złożony korzystając ze wspomnianego wcześniej polecenia **define type**. Typ składowy jest tutaj definiowany po słowie kluczowym **element,** zaś procedury wejścia i wyjścia **array_in i array_out** są w tym przypadku predefiniowane.

```
define type int4array
    (internallength=variable
    input=array_in,
    output=array_out,
    element=int4)
```

W systemie istnieje możliwość zapamiętywania wartości typu „procedura" pod postacią atrybutu, gdzie procedura jest interpretowana jako sekwencja poleceń napisanych w Postquelu – języku zapytań systemu Postgres. Umożliwia to tak zwany, konstruktor typów procedur (ang. *procedure type – constructor*). Zagadnienie to zostanie omówione szerzej w następnym punkcie.

W systemie Postgres zdefiniowano [5] ponad 30 typów predefiniowanych. Z każdym z nich jest związanych od kilku do kilkunastu operatorów i funkcji je implementujących. Jeśli dołożyć do tego możliwość definiowania przez użytkownika własnych typów, to widać wyraźnie, że wachlarz możliwości tego systemu w tym zakresie jest rzeczywiście bardzo szeroki.

Język zapytań POSTQUEL

Zestaw poleceń języka POSTQUEL to ponad 100 różnych poleceń. Ograniczymy się tutaj do omówienia tylko kilkunastu z nich, dotyczących podstawowych operacji na bazie danych, takich jak: zakładanie bazy danych, aktualizacja i wyszukiwanie danych.

Tworzenie bazy danych należy rozpocząć od wydania polecenia:

$ createdb *nazwa_aplikacji*

Użytkownik wydający to polecenie staje się automatycznie właścicielem i administratorem utworzonej aplikacji. Nazwa aplikacji musi rozpoczynać się od litery i jej długość nie może przekraczać 16 znaków. Tylko właściciel danej aplikacji może ją usunąć wydając polecenie:

$ destroydb *nazwa_aplikacji*

Istnieje kilka sposobów korzystania z danej aplikacji [2]:

- użytkownik może uruchomić interpreter poleceń systemu Postgres (w tym systemie nazywany *monitorem terminala*), który pozwala w trybie interakcyjnym na wprowadzanie, poprawianie i wykonywanie poleceń języka zapytań;
- użytkownik może kontaktować się z systemem z poziomu programu napisanego w języku C z wykorzystaniem biblioteki procedur i funkcji **LIBPQ**; pozwala to na wydawanie

poleceń języka POSTQUEL z poziomu programu, a odpowiedzi na te zapytania oraz komunikaty systemowe dotyczące statusu są przesyłane z powrotem do programu;
- użytkownik może również wykorzystywać tak zwane szybkie ścieżki dostępu (ang. *fast path*), pozwalające na bezpośrednie wykonywanie funkcji zdefiniowanych na bazie danych.

Uaktywnienie interpretera poleceń następuje po wydaniu polecenia:

$ monitor *nazwa_aplikacji*

Wydanie tego polecenia bez nazwy aplikacji powoduje próbę otwarcia domyślnej aplikacji o nazwie takiej jak nazwa użytkownika.

Na ekranie pojawia się komunikat:

```
Welcome to the C Postgres terminal monitor
GO
*
```

„Go" wskazuje, że interpreter poleceń jest gotowy do przyjmowania poleceń, gwiazdka (*) natomiast, stanowi znak zachęty (ang. *prompt*) do wprowadzania poleceń. Polecenie jest przekazywane do wykonania do jądra systemu Postgres przez wpisanie na końcu polecenia: \g, tj.

```
*polecenie \g
```

Jeśli podczas wprowadzania użytkownik popełni błąd, to w celu jego usunięcia może wywołać edytor (domyślnie **vi** z systemu UNIX), pisząc na końcu: **\e**, tj.

```
*polecenie_z_błędem \e
```

Po wykonaniu poprawek i opuszczeniu edytora poprawione zapytanie znajduje się w buforze zapytań interpretera poleceń i może być zgłoszone do systemu przez wydanie polecenia: \g

W celu zakończenia pracy z interpreterem poleceń i przejścia do systemu operacyjnego UNIX należy wydać polecenie:

```
* \q
```

W odpowiedzi, interpreter wyświetli na ekranie następujący komunikat:

```
I live to serve you
```

po czym pojawi się zgłoszenie systemu.

Pozostałe sposoby korzystania z aplikacji omówimy nieco później.

Składnia zapytania języka Postquel jest kombinacją zapytań języka Quel (z systemu Ingres) oraz SQL, i ma w wersji podstawowej postać następującą:

```
retrieve (<lista atrybutów>)
    from t1 in R1, ..., tn in Rn
    where <warunki>
```

Warunki występujące po słowie kluczowym **where** mogą być kompozycją operacji porównania wartości atrybutów, atrybutów i stałych oraz kompozycją utworzoną przez połączenie pojedynczych warunków operatorami logicznymi koniunkcji, alternatywy i negacji. Warunki tworzą tak zwaną **kwalifikację zapytania,** t_i są tak zwanymi **surogatami** (ang. *surrogates*) [2, 4] lub inaczej zmiennymi krotki, tj. elementami zastępującymi wystąpienie obiektu. Są one podstawą przeszukiwania klasy (w ujęciu tradycyjnym – relacji). Za każdym razem, gdy warunki są spełnione, jest formowana – na podstawie listy atrybutów – nowa krotka wstawiana do relacji wynikowej. Lista atrybutów tworzy tak zwaną **specyfikację klasy wynikowej** *(ang. target list)*.

Jak wspomnieliśmy wcześniej, fundamentalnym pojęciem w systemie Postgres jest pojęcie klasy. Użytkownik może utworzyć nową klasę podając jej nazwę, atrybuty i ich typy w poleceniu **create.** Na przykład, w celu utworzenia klas *pracownik* oraz *Wydziały,* należy wydać następujące polecenia:

```
* create Pracownik (nazwisko = text, placa = int4,
                    wiek = int2, wydzial = char16) \g
* create Wydzialy (nazwa = char16, kierownik = text, pietro = int2) \g
```

Wykorzystano tutaj następujące typy danych:

text – wektor znaków ASCII o zmiennej długości,

*int*4 – 4-bajtowe liczby całkowite,

*char*16 – wektor znakowy o stałej długości 16 znaków.

Wprowadzanie wystąpień do danej klasy można wykonać z wykorzystaniem polecenia **append.** Na przykład:

```
append Pracownik (nazwisko = "Z.Krolikowski", placa = 3300000,
        wiek = 39, wydzial = "elektryczny") \g
append Pracownik (nazwisko = "M.Januchta", placa = 3000000,
        wiek = 26, wydzial = "elektryczny") \g
```

Powyższe polecenia spowodują dodanie do klasy *Pracownik* dwóch nowych wystąpień. Na zdefiniowanych klasach mogą być wykonywane wszystkie podstawowe operacje relacyjne: projekcji, selekcji i połączenia. W celu wykonania operacji selekcji pracowników poniżej 35. roku życia należy wydać następujące polecenie:

```
retrieve (Pracownik. nazwisko) where Pracownik. wiek < 35 \g
```

Zwróćmy uwagę, że specyfikacja klasy wynikowej (w naszym przykładzie – *Pracownik.nazwisko*) musi być ujęta w nawiasy okrągłe.

Wynik zapytania może być opatrzony nagłówkiem. Na przykład, dla następującej kompozycji operacji projekcji i selekcji:

```
retrieve (wynik = Pracownik.placa/Pracownik.wiek)
  where Pracownik.nazwisko = "M. Januchta" \g
```

rezultat, tj. płaca i wiek pracownika *M. Januchta,* będą poprzedzone nagłówkiem **wynik.** Odpowiedź na zapytanie może być skierowana do nowej klasy w bazie danych. Kwalifikacja zapytania (tj. frazy występujące po słowach kluczowych **where** i **from**) mogą zawierać dowolną kombinację operatorów logicznych: **and, or** i **not.** W kolejnym przykładzie mamy do czynienia z sytuacją, w której wynik zapytania, tj. nazwiska pracowników, którzy nie przekroczyli 35. roku życia i zarabiają więcej niż 2 500 000 zł, zostaje skierowany do klasy *Temp* (od ang. *temporary* – tymczasowy).

```
* retrieve into Temp (Pracownik.nazwisko)
    where Pracownik.wiek < 35 and
          Pracownik.placa > 2500000 \g
```

Bardzo ciekawe możliwości stwarzają wspomniane wcześniej surogaty, tj. elementy w zapytaniu zastępujące wystąpienie w klasie. Jest to konstrukcja podobna do identyfikatora krotki lub zmiennej krotki, stosowanych w innych systemach. Dla wyjaśnienia jakie możliwości stwarzają surogaty zanalizujmy kolejny przykład. W celu wyszukania par pracowników, którzy są w tym samym wieku należy wydać następujące zapytanie.

```
* retrieve (P1.nazwisko, P2.nazwisko)
    from P1, P2 in Pracownik
    where P1.wiek = P2.wiek and P1.nazwisko != P2.nazwisko \g
```

P1 i P2 są tutaj surogatami wystąpienia w klasie *Pracownik*. W produkcie kartezjańskim dla tych wystąpień, dla których kwalifikacja zapytania jest prawdziwa, Postgres oblicza i zwraca jako wynik specyfikację klasy wynikowej.

W celu wykonania operacji aktualizacji klasy należy wykorzystać polecenie **replace.** Na przykład, chcąc zastąpić zarobki pracownika *M. Januchta* zarobkami *Z. Królikowskiego* należy wydać następujące polecenie języka POSTQUEL.

```
* replace Pracownik (Pracownik.placa = P.placa)
    from P in Pracownik
        where Pracownik.nazwisko = "M.Januchta" and
            P.nazwisko = "Z.Krolikowski" \g
```

Operację usuwania wystąpień w klasie wykonuje się z wykorzystaniem polecenia **delete**. Na przykład, polecenie

```
delete Pracownik where Pracownik.nazwisko = "M. Januchta" \g
```

spowoduje usunięcie w klasie *Pracownik* wystąpienia, dla którego wartość atrybutu nazwisko jest równa *M. Januchta*.

Jak już wspomnieliśmy, system Postgres umożliwia tworzenie złożonych typów danych – tablic jednowymiarowych o stałej lub zmiennej długości. Elementy tablic są numerowane począwszy od 1. Utwórzmy klasę *Place* zawierającą atrybuty typu tablice.

```
create Place (nazwisko = text, placa_kwartalna = int4 [4]) \g
```

Klasa *Place* zawiera dwa atrybuty: *nazwisko* – tablica znakowa o zmiennej długości, oraz *placa_kwartalna* – tablica składająca się z czterech elementów, każdy w postaci 8-znakowej liczby całkowitej. Wprowadzając dane do klasy *Place* należy pamiętać o tym, że wartości numeryczne muszą być ujęte w nawiasy klamrowe i oddzielone przecinkami.

```
*append Place (nazwisko = "Z. Krolikowski", placa_kwartalna =
    "{9000000, 9000000, 9000000, 9000000,}") \g
```

Chcąc w bazie danych wyszukać pracowników, których płace uległy zmianie w drugim kwartale, należy wydać następujące zapytanie:

```
* retrieve (Place.nazwisko)
where Place.placa_kwartalna [1]! = Place.placa_kwartalna [2] \g
```

Dużym ułatwieniem w konstruowaniu zapytań może być słowo kluczowe **all** – zastępujące wszystkie atrybuty danej klasy. Na przykład, chcąc wyświetlić zawartość klasy *Place* należy wydać następujące zapytanie:

```
* retrieve (Place.all) \g
```

Jak już wspomniano powyżej, system Postgres umożliwia korzystanie z tablic predefiniowanych w systemie, definiowanie tablic przez użytkowników oraz tworzenie tablic zawierających inne tablice. Na przykład:

```
* create Kierownik (nazwisko = text, pracownik = text[]) \g
* append Kierownik (nazwisko = "T.Morzy", pracownik =
        "{"Z.Krolikowski", "M.Januchta"}")    \g
```

Omówione powyżej polecenia języka POSTQUEL umożliwiają wykonywanie podstawowych operacji manipulowania danymi zawartymi w bazie danych. Przejdziemy obecnie do omówienia wspomnianych wcześniej elementów systemu Postgres, które odróżniają go od konwencjonalnych systemów zarządzania bazami danych. Tymi elementami są **dziedziczenie, zapytania historyczne** (ang. *time travel*) oraz rozszerzenia języka zapytań systemu definiowane przez użytkownika [7, 8].

Rozszerzenia języka zapytań POSTQUEL

Dziedziczenie

Utwórzmy w naszej przykładowej bazie danych kolejną klasę *Pracownik_na_zlecenie*.

```
* create Pracownik_na_zlecenie (nr_zlecenia = char10)
        inherits (Pracownik) \g
* append Pracownik_na_zlecenie (nr_zlecenia = "cd-23/93",
        nazwisko = "A.Kowalski", placa = 1200000, wiek = 23,
        wydział = "elektryczny") \g
```

Każde wystąpienie w klasie *Pracownik_na_zlecenie* dziedziczy (ang. *inherits*) wszystkie atrybuty (tj. *nazwisko, płaca, wiek, wydział*) z klasy nadrzędnej (inaczej klasy-rodzica) *Pracownik*. Klasa *Pracownik_na_zlecenie* ma dodatkowo atrybut *nr_zlecenia*. Zapytania mogą dotyczyć zarówno danej klasy, jak i klasy nadrzędnej. Na przykład, następujące zapytanie:

```
* retrieve (P.nazwisko)
    from P in Pracownik *
    where P.wiek >65   \g
```

spowoduje wyszukanie w klasie *Pracownik* i w *Pracownik_na_zlecenie* wszystkich wystąpień, dla których jest spełniony warunek *wiek* > 65. Gwiazdka * po nazwie klasy *Pracownik* wskazuje, że zapytanie powinno być wykonane na tej klasie i wszystkich klasach znajdujących się niżej w hierarchii dziedziczenia. W ogólności hierarchia dziedziczenia może być acyklicznym grafem skierowanym.

Zapytania historyczne

Język POSTQUEL umożliwia wykonywanie tzw. zapytań historycznych, to znaczy zapytań dotyczących stanu klasy w określonym punkcie w czasie lub przedziale czasu. Ściśle mówiąc w kategoriach temporalnych baz danych należałoby mówić o zapytaniach z mechanizmem *rollback*; są to zapytania wykorzystujące tzw. czas transakcyjny, a nie czas ważności krotki [TQuel] – rejestruje się wówczas moment wprowadzania i usunięcia krotki. Aby można było na klasie wykonywać zapytania historyczne, musi być ona utworzona z archiwizacją starych wartości. Uzyskujemy to przez polecenie **create** z użyciem opcji **archive = light** lub **archive = heavy** (dla aktualizacji przewidywanej rzadko lub często).

Chcąc uzyskać informację o aktualnych zarobkach określonego pracownika można wydać zapytanie w następującej postaci:

```
* retrieve (P.placa)
    from P in Pracownik ["now"]
    where P.nazwisko = "Z.Krolikowski" \g
```

Zauważmy, że użycie specyfikacji „now" jest równoznaczne z pominięciem jej w ogóle. Jeżeli chcemy uzyskać informacje o zarobkach tegoż pracownika w określonym przedziale czasu (np. od 1 stycznia 1982 do chwili obecnej), zapytanie to należy sformułować następująco:

```
* retrieve (P.placa)
    from P in Pracownik ["Jan 1 00:00:00 1982 GMT", "now"]
    where P.nazwisko = "Z.Krolikowski" \g
```

Przy realizacji tego typu zapytań Postgres korzysta z zegara systemowego. Czas wprowadzania i usunięcia krotki jest reprezentowany w konwencji systemu operacyjnego UNIX i zapisywany wewnętrznie w ukrytych atrybutach systemowych **tmin** i **tmax.**

Rozszerzenia języka zapytań POSTQUEL definiowane przez użytkowników

Funkcje języka C

Jednym z wielu możliwych rozszerzeń systemu jest definiowanie przez użytkownika funkcji napisanych w języku C i stosowanie ich w wyrażeniach języka POSTQUEL. Na przykład, możemy zdefiniować funkcję **wysoka_płaca,** która będzie sprawdzała, czy płacę pracownika można zakwalifikować jako wysoką.

Funkcję taką możemy wykorzystać w zapytaniu w następujący sposób:

retrieve (Pracownik.all) where wysoka_placa(Pracownik)

Pokażemy teraz, jak zdefiniować taką funkcję.

Plik o nazwie *wysoka_płaca.c* z odpowiednim programem w języku C należy skompilować używając polecenia:

```
cc -I/usr/Postgres/src/backend
   -I/usr/Postgres/src/backend/port/sparc
   -I/usr/Postgres/src/backend/obj  -c wysoka_placa.c
```

Plik korzystając z systemowych definicji typów danych (takich jak *TUPLE* czy *int4*) musi dołączyć plik nagłówkowy z deklaracjami *builtins.h*. Osiągamy to umieszczając na początku programu dyrektywę #*include* ⟨*utils/builtins/h*⟩ i wskazując dyrektywami kompilatora *-I* w jakich kartotekach kompilator powinien poszukiwać potrzebnych plików. Kartoteka *port* grupuje definicje związane z rozmaitymi architekturami komputerów, np. *sparc* oznacza procesor typu *Sparc* komputerów *Sun*. W powyższym przykładzie założyliśmy, że system Postgres został zainstalowany w kartotece */usr/Postgres*; przy innej instalacji powyższe dyrektywy powinny oczywiście zostać zmodyfikowane.

Po kompilacji powstaje plik wynikowy *wysoka_placa.o*. Możemy wówczas wprowadzić naszą funkcję do systemu wskazują, że jej kod znajduje się w wymienionym pliku:

```
define function wysoka_placa
        (language="c", returntype=bool)
        arg is (Pracownik)
        as "/staff/martin/wysoka_placa.o"
```

A oto przykładowy kod tej funkcji:

```
/* wysoka_placa.c */
#include <utils/builtins.h>
bool wysoka_placa(t)
TUPLE t;
{
  extern char *GetAttributeByName();
  short placa;
  placa = (short) GetAttributeByName(t, "placa");
  return (placa > 4000000);
}
```

Możemy zrezygnować z korzystania z definicji typów systemowych i bezpośrednio przekazywać parametry do funkcji. Kod takiej wersji funkcji **wysoka_płaca** znajduje się poniżej. Kompilacja następuje wówczas z użyciem dyrektywy:

```
cc -I/usr/Postgres
/src/backend -c wysoka_placa.c
#include <tmp/c.h>
bool wysoka_placa(placa)
int placa;
{
  return (placa>4000000)
}
```

Przy definiowaniu operatora = dla typu *box* wykorzystaliśmy funkcję *equal_area* obliczającą czy obszary prostokątów są równe. Może być ona zdefiniowana poleceniem:

```
define function-equal_area
    (language="c", returntype=bool,
    arg is (box, box)
    as "/usr/Postgres/box.o"
```

Funkcje języka POSTQUEL

W poprzednim punkcie została zdefiniowana funkcja **wysoka_płaca,** z wykorzystaniem odpowiedniej procedury napisanej w języku C. Obecnie przedstawimy inny sposób definiowania tej procedury, z wykorzystaniem poleceń języka POSTQUEL.

Dowolny zestaw poleceń języka POSTQUEL może być ujęty w formie określonej jednostki, wywoływanej i wykonywanej przez podanie jej nazwy i ewentualnie parametrów. Taki zestaw poleceń w systemie Postgres nazywa się **funkcją języka zapytań** (ang. *query language function*). Nasza przykładowa funkcja **wysoka_płaca,** w postaci przedstawionej poniżej, jako wynik dostarcza wszystkie wystąpienia klasy *Pracownik,* dla których spełniony jest warunek:
płaca > *$1*, gdzie *$1* jest nazwą parametru, którego wartość należy podać przy wywołaniu funkcji.

```
define function wysoka_placa
    (language = "Postquel", returntype = setof Pracownik)
    arg is (int4)
    as "retrieve (Pracownik.all) where Pracownik.placa > $1"
```

Jeśli w definicji funkcji występują parametry, to należy zdefiniować ich typ (dla naszego przykładu – *arg is* (*int4*)). W definicji funkcji należy również określić język, w którym funkcja została zdefiniowana (*language* = *„Postquel"*) oraz typ wyniku (*returntype* = *setof Pracownik* – co tutaj oznacza, że jako wynik wykonania funkcji uzyskamy zbiór wystąpień klasy *Pracownik,* spełniających warunki określone w definicji funkcji, tj. po słowie kluczowym **where**).

Typy złożone

Omówione wyżej funkcje można wykorzystać do budowania tzw. typów złożonych. Rozważmy możliwość rozszerzenia klasy *Pracownik* o atrybut *kierownik*. To znaczy z każdym wystąpieniem w klasie *Pracownik,* chcemy związać inne wystąpienie z tej klasy, określające kierownika danego pracownika. W tym celu należy zdefiniować następującą funkcję:

```
define function kierownik
    (language = "Postquel", returntype = Pracownik)
    arg is (Pracownik)
```

```
as "retrieve (Pracownik.all) from P in Pracownik
    where P.nazwisko = wydzialy.kierownik
    and $1.wydzial = wydzialy.nazwa "
```

Wykonanie funkcji *kierownik* możemy traktować jako dodanie do klasy *Pracownik* atrybutu złożonego typu *kierownik*, którego wartością jest wystąpienie klasy *Pracownik*. Mówimy o utworzeniu w bazie danych **typu złożonego.** Innymi słowy, funkcję *kierownik* użytkownik może traktować jak zwyczajny atrybut w klasie *Pracownik* i wykonywać na nim dowolne operacje, np.

```
* retrieve (Pracownik.nazwisko)
  where Pracownik.kierownik.nazwisko = "T. Morzy" \g
```

to znaczy, „wyszukaj wszystkich pracowników, których kierownikiem jest *T. Morzy*". Jedyny wyjątek stanowi polecenie **append** – nie można wykonać tego polecenia sformułowanego w postaci:

```
* append Pracownik (Pracownik.kierownik.nazwisko =
  = "Kowalski") \g
```

Rozważmy jeszcze jeden przykład definiowania funkcji języka zapytań. Załóżmy, że w bazie danych jest przechowywana klasa *Dzieci*, której atrybutami są dane dziecka, nazwisko ojca i matki. Zdefiniowana niżej funkcja *dzieci* „dowiązuje" do każdego pracownika jego dzieci.

```
* define function dzieci
    (language = "Postquel", returntype = setof Dzieci)
    arg is (Pracownik)
    as "retrieve (Dzieci.all)
        where $1.nazwisko = Dzieci.ojciec
        or $1.nazwisko = Dzieci.matka " \g
```

W porównaniu z poprzednią, funkcja *dzieci* dostarcza zbioru wystąpień. Zapytanie:

```
* retrieve (Pracownik.nazwisko, Pracownik.dzieci.nazwisko)
```

spowoduje wyświetlenie listy pracowników i ich dzieci.

Biblioteka funkcji systemowych LIBPQ

Oprócz wydawania poleceń poprzez interpreter poleceń, użytkownik systemu Postgres dysponuje możliwością komunikacji z systemem z poziomu własnych programów napisanych w języku C. Procedury przekazujące dane i polecenia pomiędzy procesem użytkownika a Postgresem są zawarte w bibliotece LIBPQ [3].

Program korzystający z LIBPQ musi najpierw nawiązać połączenie z systemem, wywołując procedurę **PQsetdb** i przekazując jej w argumencie nazwę aplikacji bazy danych, z której zamierza korzystać. Podstawową funkcją biblioteki jest **PQexec,** która przekazuje swój argument systemowi jako komendę do wykonania. Argumentem może być dowolne polecenie systemu Postgres. Jeżeli polecenie nie zwraca wyniku (tak jest w przypadku **append, replace** i **delete**), wynikiem poprawnego wykonania **PQexec** jest przekazanie nazwy polecenia poprzedzonej znakiem 'C' (np. CREPLACE), zaś przy błędzie jest przekazywana litera 'R'.

Jeżeli próbujemy przez **PQexec** wykonać operację **retrieve,** która zwraca określony wynik, musimy zadbać o poprawne odebranie krotek znalezionych w efekcie wykonania takiego zapytania. Przy wykonywaniu zapytań w trybie interpretera wynik jest wyprowadzany wprost na ekran, natomiast **PQexec** przekazuje znalezione obiekty do bufora komunikacyjnego, który w systemie Postgres jest nazywany **portalem.** Zapytanie **retrieve** musi być rozszerzone o podanie frazy **portal** ⟨**nazwa_portala**⟩.

Przeglądanie wyniku zapytania polega na ściągnięciu do portala określonej liczby krotek i sukcesywnym ich przeglądaniu wewnątrz portala. Wprowadzenie do portala krotek następuje w wyniku wykonania polecenia **fetch** z podaniem liczby krotek ściąganych jednorazowo. Można zażądać sprowadzenia do portala wszystkich krotek przekazywanych przez zapytanie wydając polecenie „*fetch all in nazwa_portala*". Odczytywanie danych odbywa się następnie z obszaru danych portala, do którego wskaźnik przekazuje funkcja **PWparray(nazwa_portala).**

W portalu może być kilka różnych grup krotek różniących się strukturą. Tak dzieje się w przypadku zapytań z dziedziczeniem atrybutów, gdy krotki mogą pochodzić z różnych klas powiązanych w hierarchii dziedziczenia. Do sprawdzenia liczby grup obecnych w portalu służy funkcja **PQngroups.** Dla każdej grupy możemy przeczytać nazwy kolejnych atrybutów i ich liczbę z wykorzystaniem funkcji **PQfnameGroup** i **PQnfieldsGroup.** Liczbę krotek z danej grupy obecną w portalu udostępnia funkcja **PQntuplesGroup,** zaś wartości atrybutów konkretnej kortki – funkcja **PQgetvalue.**

Dla zilustrowania powyższych operacji przeanalizujmy przykładowy program odbierający wynik zapytania „retrieve portal p1 (Pracownik.all)". Zakładamy, że wszystkie krotki należą do jednej grupy, stąd numer grupy w odpowiednich funkcjach jest zerowy.

```
#include <stdio.h>
#include "tmp/libpq-fe.h"

void main() {

int i,j,k,g,n,m,t;
PortalBuffer *p;
char pnames[MAXPORTALS][portal_name_length];

PQsetdb("martin");
PQexec("begin");    /* początek transakcji */
PQexec("retrieve portal p1 (Pracownik.all)");

PQexec("fetch all in p1");
p=PQparray("p1");  /* p wskazuje na zawartość bufora portala */
n=PQntuplesGroup(p,0); /* ilość krotek w buforze */
m=PQnfieldsGroup(p,0); /* ilość pól w krotce */

for (i=0;i<m;i++)  /* czytanie kolejnych nazw pól */
printf("%-15s", PQfnameGroup(p,0,i));
printf("\n");
for (i=0;i<n;i++) {   /*dla każdej krotki */
      for (j=0;j<m;j++)   /* dla każdego pola */
      printf("%-15s",PQgetvalue(p,i,j));   /* odczytaj ich wartość */

printf("\n");   /* koniec krotki */
}
PQexec("close p1");
PQexec("end");
PQfinish();
}
```

W drugiej części prezentacji systemu Postgres omówimy możliwości konstruowania aktywnych i dedukcyjnych baz danych z wykorzystaniem tego systemu oraz problem manipulowania obiektami złożonymi, takimi jak obrazy, zdjęcia czy zapis dźwięku.

LITERATURA

[1] *Oracle 7 Reference Manual.* ORACLE Corporation, 1993

[2] *The Postgres User Manual.* Ed. J. Rhein, G. Kemnitz and the Postgres Group, EECS Dept. University of California, Berkeley, 1993

[3] *The Postgres Reference Manual.* Ed. Postgres Group, EECS Dept. University of California, Berkeley, 1993

dokończenie na s. 21

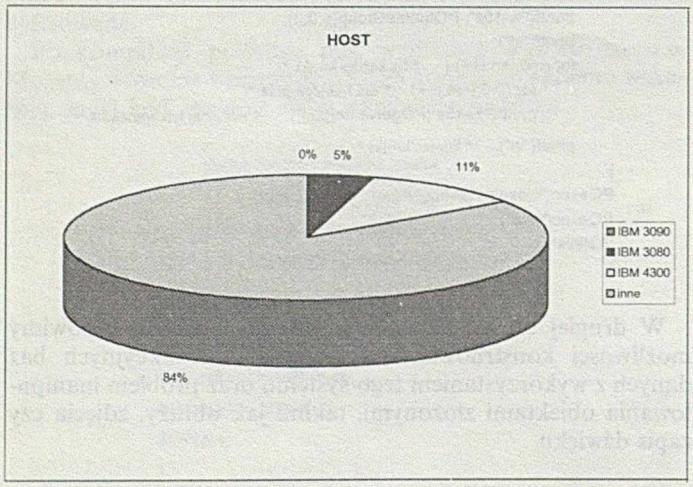
Ankieta sieciowa

Z rankingów prowadzonych od trzech lat przez redakcję ComputerWorld, wiemy jakie obroty osiągają w Polsce firmy działające na rynku komputerowym, orientujemy się z grubsza ile jest u nas komputerów i drukarek. Trudno natomiast dziś o odpowiedź na pytanie jakie, gdzie i ile mamy sieci. Tymczasem nikogo nie trzeba przekonywać, że komputery nie współpracujące ze sobą w sieci – nie są w pełni wykorzystane.

Próbę dokonania takiej oceny podjęła nasza redakcja. W majowym numerze zamieściliśmy dość obszerną ankietę, adresowaną do Czytelników, którą rozesłaliśmy także do kilku tysięcy polskich przedsiębiorstw przemysłowych, usługowych, jednostek budżetowych i naukowych. Prosiliśmy adresatów, by odpowiedzieli na 15 pytań. Zwlekaliśmy z omówieniem ankiety ponieważ praktycznie do chwili oddawania tego numeru do druku jeszcze nadchodziły odpowiedzi. W sumie otrzymaliśmy ich blisko 400, co można uznać za sukces i za reprezentatywną próbę. Połowa odpowiedzi nadeszła z przemysłu, i to z wielu dużych zakładów, m.in. z Petrochemii Płockiej, Biura Konstrukcyjnego Stoczni Gdańskiej, Kopalni Miedzi „Polkowice", Elektrowni Węgla Brunatnego Turów, Fiat Auto Poland czy Cementowni Gorażdże. Następna duża grupa – 22,3% – napłynęła od firm świadczących szeroko rozumiane usługi, do których zaliczyliśmy również i transport, w tym PKP. Prawie 19% ankiet nadeszło z placówek służby zdrowia, zaś 8,8% z placówek naukowych. Nie zdecydowaliśmy się jednak na odrębne omawianie wyników ankiety osobno w każdej z tych grup, ponieważ nie wynikały z tego żadne istotne zależności.

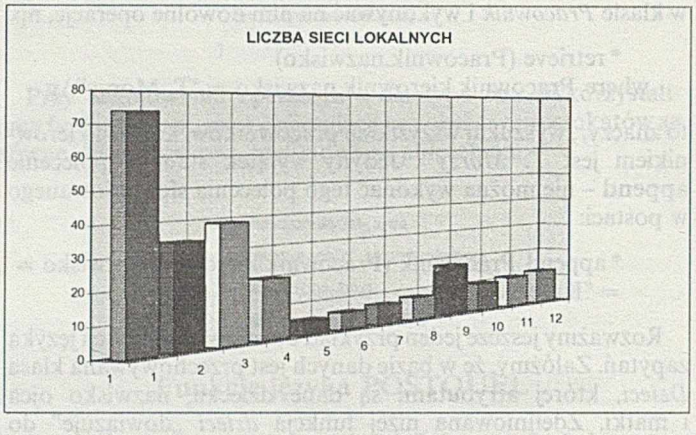
A oto odpowiedzi na pytania ankiety (w procentach). Niektóre z nich ilustrujemy dodatkowo wykresami:

1. Hosty w sieci
IMB 3080 – 5%
IBM 4300 – 11%
Inne – 84%



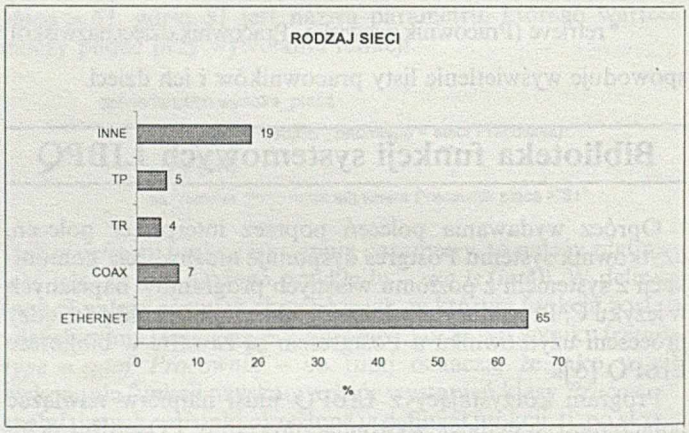
2. Minikomputery w sieci
UNIX – 17%
Wax – 15%
AS/400 – 7%
Unisys – 6%
S/3x – 6%
Inne – 49% (Wymieniane są tu m.in. MERY 9159, Odry 1305, komputery WANG)

3. Liczba sieci lokalnych
Zaledwie 50% odpowiadających na ankietę ma sieć lokalną. Z tego 75% ma jedną sieć. Pozostałe dane ilustruje wykres „Liczba sieci lokalnych".



4. System zarządzania w sieci
PATHWORKS – 16%
IBM LAN Server – 14%
LAN Manager – 8%
Inne – 62% (Pojawiają się tu sporadycznie XENIX, HP-Ux, DECnet)

5. Rodzaj sieci
Ethernet – 65%
COAX – 7%
TP – 5%
Token Ring – 4%
Inne – 19%



6. System
Novell – 69%
IBM – 13%
Inne – 18% (Wymieniono m.in. HP-UX, SCO Unix, Intergraph)

7. Oprogramowanie emulacji stacji roboczej
IBM PC/3270 – 31%
Attachmate – 2%
Rumba – 2%
Inne – 65%

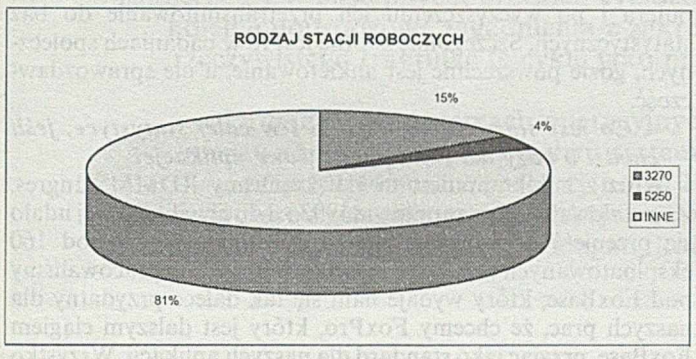
8. Liczba PC włączonych do sieci
Minimalna liczba PC-2, maksymalna – 200

9. System operacyjny PC
DOS – 73%

Windows – 17%
OS/2 – 2%
UNIX – 8%
MAC – 0

10. Liczba stacji roboczych w sieci
Minimalna – 2, maksymalna – 200

11. Rodzaj stacji roboczych
3270 – 15%
5250 – 4%
Inne – 81% (Wymieniono m.in. WYSE 120, Intergraph, Compaq Proliant 4000, HP 9000/700, Sun Sparc Station)



12. Liczba użytkowników PC lub stacji roboczych
Minimalna – 4, maksymalna – 870 (średnio wypada nieco ponad 67 użytkowników sieci)

13. Połączenie z siecią zewnętrzną
Tak – 15,3%
Nie – 84,7%

14. Typ połączenia
Linia punkt za punktem – 30%
X25 – 21%
POLPAK – 11%
Inne – 38%

15. Okres planowania inwestycji
6–12 miesięcy – 85%
3–6 miesięcy – 15%

Wyniki ankiety nie będą chyba dla nikogo zaskoczeniem. Przez wiele lat kupowano u nas komputery bez zastanawiania się nad tym czemu i jak mają służyć. Często pojedyncze komputery stały niewykorzystane lub wykonywano na nich jakieś drobne prace nie mające wpływu na funkcjonowanie firmy. Dopiero w ostatnich 2–3 latach zaczęto u nas na serio mówić o budowie sieciowych systemów informatycznych. Nie zawsze da się do ich budowy wykorzystać już posiadany sprzęt, często przestarzały moralnie i fizycznie. Sytuacja jednak zmienia się na lepsze, jak widać chociażby z odpowiedzi na ostatnie pytanie ankiety dotyczące okresu planowania inwestycji – nie aż tak szybko, by wyniki ankiety przestały być aktualne w najbliższych miesiącach. I jak byśmy sobie życzyli.

Ankietę przygotowała i opracowała: Krystyna Karwicka

Sequent i Unisys rozszerzają porozumienie OEM

Unisys Corporation jest firmą mającą ponad 60 000 odbiorców w ponad 100 krajach. Jest głównym dostawcą produktów i usług informatycznych dla sektora finansowego, administracji państwowej, telekomunikacji, transportu, służby zdrowia oraz innych dziedzin.

Sequent Computer Systems, Inc. jest czołowym dostawcą architektury otwartych systemów typu klient-serwer dla aplikacji polegających na przetwarzaniu transakcji w trybie on-line, wspomaganiu decyzji oraz przesyłaniu informacji. Sequent prowadzi również profesjonalne doradztwo i usługi szkoleniowe. Firma produkuje i sprzedaje głównie skalowalne systemy komputerowe, wspomagające aplikacje przemysłowe i usługi informacyjne: serię Symmetry z systemem operacyjnym UNIX oraz rodzinę WinServer, opartą na systemie operacyjnym Microsoft Windows NT. Od czasu wprowadzenia na rynek w 1984 r. swojej rodziny symetrycznych wieloprocesorowych komputerów Sequent zainstalował na całym świecie 6000 dużych systemów tego typu.

20 września br. firmy Sequent Computer Systems oraz Unisys Corporation zawarły trzyletnie porozumienie OEM, rozszerzające dotychczasową formę współpracy, w ramach którego Unisys będzie zwiększał efektywność swojej linii serwerów opartych na systemach Sequenta. Wchodzące w życie od 1 października br. porozumienie zapewnia zwiększenie zakresu planowania oraz rozszerzenie współpracy w dziedzinie sprzedaży.

Nowe wysokowydajne komputery Unisys'a rodziny U 6000/600 są wieloprocesorowymi symetrycznymi systemami o szerokim zakresie skalowalności (SMP). Oparte są na unixowych serwerach Sequenta serii Symmertry 5000, zawierających od 2 do 30 procesorów Pentium 66 MHz i do 4 komputerów w układzie cluster. Rodzina U 6000/600 łączy w sobie inne serwery Sequenta sprzedawane przez Unisys, z serwerami SMP: U 6000/72P, U 6000/75P i U 6000/85P oraz unixowymi serwerami Unisysa. Stanowi kombinację komputera o mocy klasy mainframe i skalowalności stacji roboczej dla aplikacji przemysłowych typu klient-serwer o szczególnie wysokim stopniu bezpieczeństwa danych.

„Nowe porozumienie OEM powiększy zakres działalności Sequenta na rynkach, na których Unisys ma rozwiniętą infrastrukturę dystrybucji, serwisu i wspomagania użytkownika, a także długoletnie związki z klientami" – powiedział John McAdam, wiceprezes ds. światowej działalności Sequenta. „Szersza współpraca między naszymi kanałami dystrybucji zapewni efektywne wspomaganie oraz przygotuje pole dla bardziej rozległej penetracji rynku, znajdującego się poza właściwą sferą działalności Sequenta. Porozumienie zapewni także wyższy poziom jednorodności systemów Sequenta sprzedawanych przez obie firmy."

„Długoletni związek Unisysa z Sequentem przyniósł wymierne korzyści odbiorcom Unisysa, a nowe porozumienie jeszcze je zwiększy. Nasze wspólne zaangażowanie się w technologię Unix i Intel zapewni podstawę, na której nasi klienci będą mogli rozwinąć własne strategie technologii informacyjnej tak, aby sprostać nie tylko dzisiejszym, ale i jutrzejszym wymaganiom" – powiedział Frank G. Brandenberg, wiceprezes Unisys Corporation i prezes ds. systemów typu klient-serwer.

Określone przez zawarte porozumienie współdziałanie kanałów dystrybucji wymaga ujednolicenia strategii obu firm na poziomie regionalnym, celem zmaksymalizowania sprzedaży ich systemów. Nowy model kooperacji międzyregionalnej, w zakresie sprzedaży dowiódł swojej skuteczności w ubiegłym roku przy umowach z klientami ze Stanów Zjednoczonych, Wielkiej Brytanii i Azji.

Oprócz istniejącego już wspomagania marketingowego i technicznego, nowe porozumienie zapewnia również większe wsparcie przez Sequent sprzedaży na poziomie lokalnym przez powołanie menedżera ds. rozwoju kanałów dystrybucji oraz podejmowanie wspólnych działań z innymi dostawcami.

(mw)

Szanse i zagrożenia

dokończenie ze s. 2

poza statystyką. Takie podejście do sprawy wynika m.in. z faktu, że na podstawie tych samych danych mogą być realizowane różne cele. Statystyka też, z natury rzeczy, nie jest zainteresowana danymi bardzo szczegółowymi, chociaż muszą być one włączone do zbiorów informacji, które pozwolą na statystyczne wnioskowanie. Ale my nie udostępniamy danych jednostkowych, również gospodarczych. Rząd i ministerstwa, które były przyzwyczajone, że dostają z GUS informację o każdym przedsiębiorstwie, zostały od tej informacji odcięte. Powstanie systemów informacji w resortach szalenie ułatwi nam pracę. W projekcie naszej ustawy jest zapis, że statystyka ma nieograniczony dostęp do zbiorów resortowych, z prawem wykorzystywania ich do badań statystycznych. Mamy nadzieję, że rodzący się system informacyjny państwa będzie logiczny, spójny, zintegrowany, funkcjonalnie poprawny, najtańszy i pozwoli na wyciąganie prawidłowych wniosków na każdym szczeblu decyzyjnym, u każdego użytkownika. Na spotkaniu w Łańsku (we wrześniu br.) pan Premier stwierdził m.in., że jest przekonany do potrzeby oparcia się na tych samych metadanych we wszystkich systemach informacyjnych administracji państwowej.

– *Jak GUS ocenia stan prac nad systemem informacyjnym administracji państwowej?*

– Mamy już zaczątki tego systemu. W GUS prowadzony jest REGON – rejestr jednostek gospodarczych. Rozpoczęły się prace związane z TERYT-em – rejestrem terytorialnym, który wymaga uwzględnienia 3 kryteriów: podziału administracyjnego, geodezyjnego i statystycznego – niezbędnego m.in. do prowadzenia badań masowych. GUS od wielu lat utrzymuje i aktualizuje rejestry podziału administracyjnego i statystycznego. Są natomiast problemy z aktualizowaniem rejestrów geodezyjnych. Muszą być powiązania między rejestrami. REGON np. musi być powiązany z PESELEM, ponieważ wiele osób fizycznych też prowadzi działalność gospodarczą.

– *Technologie informatyczne umożliwiają nie tylko gromadzenie informacji, ale i ich wykorzystywanie przeciwko społeczeństwu. Znakomicie pokazał to Orwell w „Roku 1984". Czy nie jest to realne zagrożenie?*

– Statystyka jest zobligowana do przestrzegania wszystkich obowiązujących w kraju przepisów dotyczących tajemnicy państwowej, handlowej, służbowej i ochrony dóbr osobistych. Mimo braku niektórych uregulowań prawnych – ustawy o ochronie dóbr osobistych – sami dbamy o to, by swoimi poczynaniami nie doprowadzać do sytuacji amoralnych społecznie. Obchodziliśmy w ub.r. 75-lecie GUS i 200-lecie istnienia statystyki; ani razu nie zdarzyło się, aby w tym okresie złamano zasadę nieudostępnienia danych osobistych.

– *A dobry hacker nie może dotrzeć do nich?*

– W zbiorach nie przechowujemy danych identyfikowalnych, tzn. nie wprowadzamy żadnych imion własnych. Pomijam tu rejestry stanowiące osobne zbiory, w których nie ma żadnych danych statystycznych.

– *W jaki sposób są archiwizowane dane, ile ich jest i jak docierają do GUS?*

– W centralnej bazie danych, która obejmuje tylko informacje z dziedziny statystyki ekonomicznej, jest ok. 20 GB. W REGONIE ok. 15 GB. Nie więcej niż 5% dociera do nas z innych systemów. Jedynie niewielką ich część otrzymujemy w sposób zautomatyzowany. Dane docierają do nas na arkuszach statystycznych, następnie musimy je wprowadzać do bazy. Zadaniem podstawowym Wojewódzkich Urzędów Statystycznych jest zebranie sprawozdań, wprowadzenie danych i doprowadzenie zbioru do poprawności z punktu wymaganych kryteriów. WUS-y przekazują zbiór transmisyjnie do Warszawy. Ten proces można byłoby zgrabnie zautomatyzować korzystając z czytników dokumentów. Na razie nas na to nie stać. Obecnie skanery wykorzystujemy jedynie do publikacji. Nasze archiwa oparte są na technologii taśm magnetycznych, zrobił się też gąszcz dyskietek, nad którymi nikt nie jest w stanie zapanować. Dlatego w resorcie statystyki zatrudnionych jest tak dużo osób. Żeby statystycy mogli lepiej korzystać z bazy i z udogodnień, jakie stwarzają narzędzia informatyczne – konieczna jest sieć. Zaczęto od kładzenia sieci w budynku głównym, powinna być gotowa pod koniec I kw. 1995 r. Podobne zadania są realizowane w części WUS-ów. Dość powszechną praktyką na Zachodzie jest wprowadzanie danych na miejscu do podręcznego komputera i po wyczyszczeniu ich przetransmitowanie do baz statystycznych. Szczególnie istotne jest to w badaniach społecznych, gdzie powszechne jest ankietowanie, a nie sprawozdawczość.

– *Co jest standardem w GUS i w całej statystyce, jeśli chodzi o bazy danych i podstawowe aplikacje?*

– Wraz z minikomputerami HP kupiliśmy RDMMS Ingres. Zainstalowaliśmy go w połowie 1993 r. i do chwili obecnej udało się przenieść pod Ingresa ponad 30 aplikacji z pośród 160 eksploatowanych w naszym resorcie. Poprzednio pracowaliśmy pod FoxBase, który wydaje nam się tak dalece przydatny dla naszych prac, że chcemy FoxPro, który jest dalszym ciągiem FoxBase, przyjąć jako standard dla naszych aplikacji. Wszystko jest podporządkowane systemom otwartym, czyli obecnie Unix i Windows NT. Standardem edytora tekstu jest WordPerfect, arkusza kalkulacyjnego – Lotus.

– *Czy tak znaczne uzależnienie statystyków od informatyków i ich narzędzi technologicznych nie powoduje konfliktów między nimi?*

– Nie dostrzegam konfliktu interesów. Chciałabym natomiast ostrzec, że może zabraknąć informatyków w statystyce. Poziom płac w GUS jest niższy w stosunku do budżetowych jednostek, nie wspomnę o ministerstwach. GUS został dzięki temu pozbawiony kadry informatycznej o średnim stażu pracy. A tymczasem mamy realizować duże zadania modernizacyjne, przyuczyć wiele osób do korzystania z narzędzi informatycznych i realizować normalną produkcję. To poważne zagrożenie, mimo iż GUS od strony profesjonalnej jest konkurencyjny. Chciałabym też wspomnieć o innym niebezpieczeństwie, związanym z niedocenianiem w Polsce umiejętności wykorzystywania metod statystycznych nie tylko przez statystyków,. Umiejętność ta pozwala na poprawne wyciąganie wniosków z określonych zasobów niezbędnych informacji. Jest ona niezbędna tym wszystkim, którzy chcieliby podejmować sensowne decyzje. Tymczasem u nas zmniejsza się liczba godzin nauczania statystyki zarówno na uczelniach, jak i w szkołach średnich. Zredukowano, bądź wręcz zlikwidowano kierunki o specjalizacji statystycznej w wielu placówkach dydaktycznych. Samo zebranie informacji w jakimś zbiorze i ustalenie reguł jego udostępnienia – co dziś jest możliwe właśnie dzięki informatyce – to jeszcze nie statystyka, a jedynie przetwarzanie. Nawet najlepszy zbiór informacyjny będzie bezwartościowy, albo doprowadzi do niewłaściwych wniosków, jeśli człowiek z niego korzystający nie będzie posiadał dostatecznej wiedzy, by się nim posłużyć.

– *Dziękujemy Pani za rozmowę.*

Do druku przygotowała: Krystyna Karwicka

Sprostowanie

Przepraszamy za błędne podanie na s. 6 sierpniowego numeru nazwy banku, w którym jest konto Fundacji Pomocy Niepełnosprawnym Ruchowo. Powinno być:

Bank Gospodarki Komunalnej
I/O Warszawa nr konta: 399003-10802047-2710.

SYSTEMY CZASU RZECZYWISTEGO

REDAGUJE ZESPÓŁ: dr inż. Zbigniew FRYŹLEWICZ (Politechnika Wrocławska), prof. dr hab. Janusz GÓRSKI (Francusko-Polska Wyższa Szkoła Nowych Technik Informatyczno-Komunikacyjnych w Poznaniu), prof. dr hab. inż. Zbigniew HUZAR (Politechnika Wrocławska), dr inż. Krzysztof SACHA (Politechnika Warszawska), prof. dr hab. inż. Tomasz SZMUC (Akademia Górniczo-Hutnicza), dr inż. Zbigniew ZIELIŃSKI (Wojskowa Akademia Techniczna), mgr inż. Zdzisław ŻURAKOWSKI (Fabryka Wagonów PAFAWAG we Wrocławiu) – przewodniczący zespołu.

ROK I ● Listopad ● 1994 ● 2

W bieżącym numerze kontynuujemy rozpoczęty w numerze 1 Wkładki cykl artykułów poświęcony systemom operacyjnym czasu rzeczywistego. Po zakończeniu tego cyklu, zamierzamy w następnych, krótkich cyklach tematycznych (po 3–4 artykuły) zaprezentować kolejne podstawowe zagadnienia z zakresu projektowania i eksploatacji systemów czasu rzeczywistego traktując te cykle jako przystępne wprowadzenie w zagadnienie inżynierii systemów czasu rzeczywistego.

Drugim – po systemach operacyjnych czasu rzeczywistego – będzie cykl poświęcony niezawodności i bezpieczeństwu systemów komputerowych, w ramach którego chcemy przedstawić m.in. projekty norm IEC dotyczących oprogramowania dla systemów komputerowych pełniących funkcje związane z bezpieczeństwem. Trzeci cykl poświęcimy metodom analizy i specyfikacji wymagań, kolejny – metodom projektowania systemów czasu rzeczywistego. (Z.Ż).

Synchronizacja w rozproszonych systemach uwarunkowanych czasowo

Elżbieta Kosmulska-Bochenek
Centrum Informatyczne
Politechnika Wrocławska

Dla wielu aplikacji systemów komputerowych wystarczającą miarą efektywności jest czas odpowiedzi (ang. *response time*) lub czas obrotu zadania (ang. *turnaround time*) bez narzucania określonych wymagań odnośnie zachowania się systemu w czasie. Dla aplikacji pracujących w czasie rzeczywistym uwarunkowania czasowe w trakcie przetwarzania mają znaczenie krytyczne.

W aspekcie ograniczeń czasowych rozróżnia się dwa rodzaje systemów czasu rzeczywistego. Pierwszy to taki, w którym uwarunkowania czasowe są na tyle istotne, że niespełnienie ich jest traktowane jako awaria systemu i wtedy mówi się o systemach czasu rzeczywistego silnie uwarunkowanych czasowo (ang. *hard real-time systems*). Przykładem może tu być dowolny system sterowania procesem technologicznym. Drugi rodzaj, to systemy czasu rzeczywistego słabo uwarunkowane czasowo (ang. *soft real-time systems*), czyli takie, w których realizacja wymagań czasowych jest tylko pożądana, a ich niespełnienie nie prowadzi do upadku systemu. Jednym z przykładów takich systemów jest aplikacja przetwarzania transakcji.

Oba typy systemów czasu rzeczywistego pracują najczęściej w środowisku rozproszonym, a dodatkowo, jeśli dotyczą sterowania urządzeniami, przetwarzanie odbywa się w trudnych warunkach otoczenia. Takie warunki pracy narzucają wiele cech, jakie powinien mieć system operacyjny wspomagający funkcjonowanie aplikacji w czasie rzeczywistym.

Naturalnymi wymaganiami, wynikającymi z charakteru środowiska, w jakim systemy te pracują, stawianymi systemom operacyjnym czasu rzeczywistego, są wielozadaniowość i wieloprocesorowość. Oprogramowanie jest skonstruowane na ogół w postaci zadań, które obsługują asynchroniczne zdarzenia zachodzące w systemie rzeczywistym (np. w sterowanym procesie technologicznym). Rolą systemu operacyjnego jest zapewnienie prawidłowej obsługi tych zdarzeń.

Wyróżnia się dwa podejścia do sposobu traktowania zdarzeń w systemie: obsługa na podstawie priorytetów [3, 4, 7] oraz obsługa na podstawie czasu [5] Obie koncepcje mają swoje wady i zalety. U podstaw rozwiązania zagadnienia czasu metodą obsługi zdarzeń – zgodnie z ich priorytetami – leży naturalne założenie, że procesy pracujące w czasie rzeczywistym mają na ogół przypisane im priorytety, stąd uszeregowanie zadań powinno wynikać z priorytetów procesów a nie czasu ich wystąpienia. Podejście to, historycznie starsze, wpływa korzystnie na efektywność całego systemu [2], a poza tym nie wymaga organizowania w nim czasu globalnego.

Za sterowaniem zdarzeń na podstawie czasu przemawia fakt, że poprzednie rozwiązanie nie uwzględnia limitu czasu, w jakim proces ma być obsłużony (ang. *deadline*), czego w wielu zastosowaniach nie można pominąć. Ponadto wiele aplikacji wymaga obsługi zdarzeń w ściśle określonej chwili, czego nie uzyska się przez sterowanie priorytetami.

Ze względu na podwyższoną niezawodność oraz częste rozproszenie elementów procesu technologicznego, systemy te mają strukturę modularną (wieloprocesorową), co w sposób naturalny narzuca rozproszoną lub sieciową postać systemu operacyjnego.

Artykuł stanowi nieco zmienioną wersję referatu, wygłoszonego na Ogólnopolskiej Konferencji Naukowo-Technicznej pn. „Oprogramowanie komputerowych systemów czasu rzeczywistego – zagadnienia związane z wytwarzaniem oprogramowania do zastosowań przemysłowych", która odbyła się we Wrocławiu 22–23.09.1994 r.

Podstawowa różnica między systemami jednoprocesorowymi i rozproszonymi polega na sposobie komunikacji między procesami. W systemach jednoprocesorowych procesy w większości przypadków korzystają ze wspólnej pamięci, gdzie informacja może być przekazana do bufora przez jeden proces i pobrana z niego przez inny proces. Systemy operacyjne dostarczają odpowiednich mechanizmów wzajemnego wykluczenia w dostępie do sekcji krytycznej oraz synchronizacji współdziałania procesów. W systemach rozproszonych procesy komunikują się przez przesyłanie komunikatów, stąd zagadnienia synchronizacji i wzajemnego wykluczenia mają nieco inny charakter. Decydują o tym takie czynniki, jak:

- rozproszenie informacji między wieloma węzłami sieci,
- podejmowanie decyzji w poszczególnych węzłach na podstawie lokalnie dostępnej informacji,
- fakt, że nie istnieje pojedyncze źródło uszkodzenia, tzn. system nie ulega awarii w wyniku uszkodzenia jednego węzła,
- brak globalnego czasu w systemie.

Pierwsze trzy wymienione pozycje mają wpływ na niezawodność takiego systemu, która z założenia musi być wyższa niż systemu jednoprocesorowego. Zdecydowany wpływ na sposób pracy rozproszonego systemu ma brak globalnego czasu. Każdy węzeł (komputer) ma swój czas lokalny, który nie może być podstawą do jakichkolwiek uwarunkowań współpracy procesów w całym systemie. Z drugiej strony wiadomo, że w systemach czasu rzeczywistego czas odgrywa decydującą rolę zarówno przy planowaniu procesów, jak i kontroli krytycznych czasów obsługi zdarzeń (ang. *deadline*). Również przy braku wspólnej pamięci, globalny czas jest podstawą do prawidłowej realizacji komunikacji międzyprocesowej, a w szczególności synchronizacji i wzajemnego wykluczenia.

Pojęcie synchronizacji odnosi się do trzech powiązanych ze sobą zagadnień:

- synchronizacja między nadawcą i odbiorcą wiadomości,
- specyfikacja i sterowanie wspólnymi aktywnościami współpracujących ze sobą procesów,
- serializacja równoczesnego dostępu wielu procesów do współdzielonych obiektów.

W rozproszonych systemach czasu rzeczywistego, synchronizacji nie można uzyskać bez wymiany wiadomości między procesami, a dodatkowo w tego rodzaju systemach w każdym kroku pracy pojawiają się nowe zdarzenia, lokalne dla jednego procesu, a nieprzewidziane dla innych. Dla takiego modelu systemu należy przyjąć, że pod pojęciem synchronizacji rozumie się czasowe uporządkowanie zbioru zdarzeń generowanych przez procesy równocześnie przetwarzające się w czasie.

Zegary

Niemal każdy komputer wyposażony jest w zegar. Podstawowym jego elementem jest kryształ kwarcu, który zależnie od rodzaju i sposobu oszlifowania generuje sygnał o określonej częstotliwości. Mimo dużej stabilności oscylacji kryształów niemożliwe jest uzyskanie dwóch zegarów generujących czas z taką samą częstotliwością. W praktyce, jeśli system składa się z n komputerów, każdy z n kryształów pracuje z różną szybkością, powodując z czasem między nimi coraz większe różnice. Odchylenia te (ang. *skew*) stanowią duże utrudnienie w prawidłowej komunikacji między procesami, a w przypadku krytycznego uwarunkowania czasem, jakie występuje w systemach pracujących w czasie rzeczywistym, wręcz ją uniemożliwia. Istotna więc jest taka synchronizacja wszystkich zegarów w systemie, by uzyskać w nim jeden, unikatowy czas standardowy. Nie jest to zadanie łatwe ze względu na połączone rozliczne typy komputerów, dodatkowo utrudnione przez opóźnienia uzyskiwane na liniach telekomunikacyjnych.

Dla wielu aplikacji nie jest konieczna bezwzględna synchronizacja czasu – jeśli procesy nie komunikują się, nie ma potrzeby synchronizować ich zegarów. Często dla współpracujących procesów nie jest też istotna wartość czasu, natomiast istotna jest kolejność występowania zdarzeń. W innych przypadkach może być wystarczające uzgodnienie między komputerami wartości czasu, który nie musi pokrywać się z rzeczywistym, uzyskanym ze wzorca. W ten sposób utrzymuje się spójność czasu, a także zorganizowane zegary noszą nazwę **zegarów logicznych** w odróżnieniu od **zegarów fizycznych,** które charakteryzują się dodatkowym założeniem o nieprzekraczalności maksymalnej wielkości odchylenia od czasu rzeczywistego.

Wyróżnia się trzy kategorie systemów zegarowych:

- centralne,
- sterowane centralnie,
- rozproszone.

Systemy centralne charakteryzują się tym, że jeden zegar (proces) jest źródłem dokładnego czasu dla całego systemu, a w przypadku jego awarii funkcję tę jest w stanie podjąć każdy inny węzeł. Procesy w systemie mogą uzyskać czas przez odczyt lub przez komunikat. Metoda implementacji czasu przez odczyt wymaga dodatkowo wspomagania sprzętowego procesora, by ustawioną przez zegar centralny wartość czasu mógł przeczytać każdy proces (jeśli czas nie jest przesyłany), co podraża koszty metody. Natomiast bezsporne jej zalety to niezawodność (potencjalna gotowość innych zegarów) oraz niski koszt komunikacji (jedna wiadomość wystarczy do synchronizacji).

W systemach sterowanych centralnie wyróżnia się dwa rodzaje węzłów: główne (ang. *master*) i podrzędne (ang. *slave*). Desygnowany węzeł typu *master* kontroluje czas w podległych sobie węzłach typu *slave*. Jeśli zegar główny ulegnie awarii, funkcję jego przejmuje, w wyniku wyboru, jeden z węzłów typu *slave*. Taki system wymaga obliczenia różnic czasu, wynikających z opóźnień transmisji i uwzględniania poprawek w stosunku do otrzymanej wartości czasu.

Rozproszony system zegarowy polega na realizacji w każdym węźle takiego samego algorytmu przekazywania czasu innym węzłom i uaktualnianiu swojego bieżącego czasu. Metoda ta daje duże natężenie ruchu w sieci, ponieważ podstawą jej jest aktywność każdego węzła w sieci, w wyniku której informacja o wielkości czasu rozprzestrzeniana jest w całym systemie. Znanych jest kilka algorytmów synchronizacji zegarów, opartych na rozproszonych systemach zegarowych [6]. Wszystkie one opierają się na takim samym modelu.

Zakłada się, że każdy komputer ma własny lokalny zegar C. Jeśli oznaczyć czas wzorcowy jako t, to wartość czasu podawaną przez zegar w maszynie p określa się jako $Cp(t)$. Oczywiście, dla warunków idealnych byłoby $C(t) = t$ dla wszystkich p i wszystkich t, innymi słowy, byłoby spełnione, że $dC/dt = 1$. W warunkach rzeczywistych zależność ta przybiera postać:

$$1 - \sigma \leqslant dC/dt \leqslant 1 + \sigma$$

gdzie σ jest stałą określoną przez producenta, charakteryzującą maksymalną szybkość odchylania się od ustawionej wartości (ang. *maximum drift rate*). Gdy $dC/dt > 1$, zegar „spieszy", a gdy $dC/dt < 1$, zegar opóźnia się. Jeśli dwa zegary w systemie, po zsynchronizowaniu, dryfują w przeciwnych kierunkach, różnica w ich wskazaniach powiększa się z czasem. System operacyjny musi więc zapewnić właściwą częstotliwość powoływania algorytmu synchronizacji w celu utrzymania dokładnego i stabilnego czasu w systemie. Jeśli założyć, że dwa zegary w systemie nie mogą różnić się więcej niż

dozowolona wartość d, to zegary muszą być uaktualniane przynajmniej co $d/2\sigma$ sekund.

Synchronizacja zegarów fizycznych

Systemy wielozegarowe z jednej strony spełniają wysokie wymagania niezawodnościowe stawiane systemom czasu rzeczywistego oraz podwyższają ich efektywność, a z drugiej strony takie rozwiązanie niesie ze sobą nowe problemy. Po pierwsze, jak zsynchronizować zegary z wzorcowym czasem rzeczywistym, po drugie, jak zsynchronizować je razem. Jeśli założyć, że uzyskanie wzorcowego czasu rzeczywistego (sygnał radiowy lub satelitarny) nie stanowi problemu i jeden komputer systemu jest przystosowany do odbioru takiego czasu, to zagadnieniem do rozwiązania pozostaje w takim systemie tylko synchronizacja zegarów. Jeśli żaden komputer nie odbiera czasu wzorcowego, synchronizacja takiego systemu polegać będzie na utrzymaniu minimalnych, jak tylko to możliwe, różnic między lokalnymi wartościami czasu w poszczególnych węzłach systemu.

Centralne systemy zegarowe

W systemie znajduje się źródło czasu wzorcowego (serwer czasu). Okresowo każdy komputer wysyła do serwera zapytanie o czas. W odpowiedzi, możliwie szybko, przekazywany jest komunikat zawierający bieżący czas. Otrzymana wartość z serwera staje się podstawą do aktualizacji czasu każdego komputera w systemie.

Takie rozwiązanie ma dwie niedogodności. Pierwsza z nich to sytuacja, kiedy wartość czasu zegara nadawcy jest wyższa od wartości otrzymanej z serwera. Nie jest możliwe cofnięcie czasu, ponieważ skutki takiego działania byłyby katastrofalne dla systemu. Pozostaje więc zastosowanie metody stopniowego uaktualniania czasu, polegającej na dodawaniu za każdym kolejnym przerwaniem jednostki mniejszej niż normalnie (tzn. bez aktualizacji). Wynika to ze sposobu pracy systemu, np. jeśli zegar generuje 100 przerwań na sekundę i każde przerwanie uaktualniałoby czas co 10 ms, to wartością aktualizacyjną będzie 9 (jeśli zegar jest spowalniany). Podobnie sposób ten może być zastosowany przy stopniowym, zamiast skokowym, przyspieszaniu zegara – dla omawianego przykładu, będzie to dodanie wartości 11.

Drugą niedogodność stanowi opóźnienie uzyskanej odpowiedzi, wynikające z czasu obsługi wiadomości w serwerze i opóźnienia transmisji. Jeśli znane są takie dane, jak: czas obsługi wiadomości w serwerze S, czas wysłania zapytania to oraz otrzymania odpowiedzi t_1 (te dwa ostatnie są lokalnymi wartościami czasu w jednym komputerze), to poprawkę można oszacować jako $(t_1 - t_0 - S)/2$. Przy braku danych o czasie transmisji i obsługi przerwań w serwerze oszacowanie będzie wystarczająco dobre, jeśli uwzględnić tylko dane lokalne, to znaczy przyjąć czas propagacji jako $(t_1 - t_0)/2$.

Sterowanie centralne

W poprzednim rozwiązaniu serwer był pasywny oraz istniało w systemie źródło czasu wzorcowego. Innym podejściem jest synchronizacja czasu przy założeniu aktywnego serwera, umożliwiająca synchronizację w systemie bez takiego źródła. Okresowo serwer wysyła zapytanie do wszystkich komputerów o ich bieżący czas. Na podstawie uzyskanych odpowiedzi serwer oblicza średni czas i rozsyła tak uzyskany czas do komputerów w sieci.

Funkcjonowanie algorytmu ilustruje rys. 1. Proces serwera odczytuje własny czas t i rozsyła go do węzłów (procesów) o lokalnych czasach $t1$ i $t2$ (1a). Węzły te odsyłają serwerowi różnicę między własnym czasem, a czasem otrzymanym (1b). Proces serwera na tej podstawie oblicza średnią wartość czasu i rozsyła podległym węzłom wartość (dodatnią lub ujemną), o jaką procesy muszą zaktualizować własny czas (1c).

Systemy rozproszone

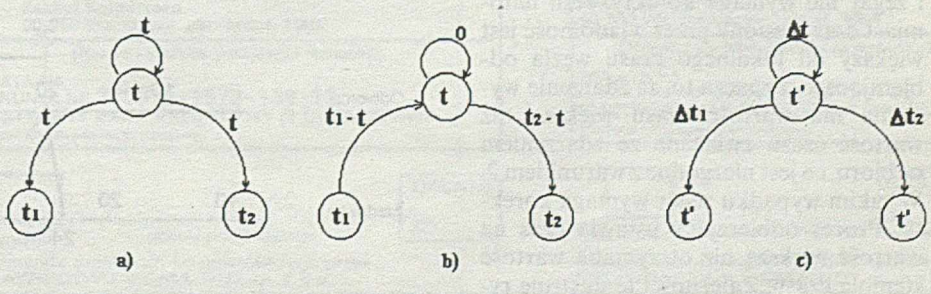
Algorytmy tej klasy polegają na rozgłaszaniu przez każdy węzeł w określonych interwałach czasu, wartości bieżącego czasu, zgodnie ze swoim lokalnym zegarem. Ponieważ zegary w różnych komputerach nie pracują dokładnie z taką samą szybkością, rozgłaszanie takie nie będzie odbywało się dokładnie w tym samym czasie. Po przekazaniu własnego czasu, węzeł gromadzi wszystkie wartości czasów nadesłane przez inne węzły w tym przedziale. Po zebraniu wszystkich danych, powoływany jest algorytm obliczania nowego czasu. Najprostszym z nich jest uśrednienie uzyskanych ze wszystkich komputerów wartości czasu. Innym sposobem może być odrzucenie wartości ekstremalnych i uśrednienie pozostałych. To ostatnie rozwiązanie równocześnie wyklucza pobranie do obliczeń wartości przypadkowych, uzyskanych od uszkodzonych zegarów.

Dokładność oszacowania czasu można zwiększyć przez uwzględnienie w obliczeniach dodatkowo czasu propagacji wiadomości z jej źródła. W tym ostatnim przypadku wymaga się jeszcze znajomości topologii sieci oraz czasu transmisji między węzłami. Przykładem takiego algorytmu może być algorytm opisany w [1], gdzie każda jednostka jest połączona z inną lub z grupą innych jednostek, którym wysyła wiadomość. Na podstawie tej wiadomości lokalne jednostki uaktualniają własny czas łącznie z uwzględnieniem odległości. Metoda ta, zwana prostą dyfuzją zegarową (ang. *simple clock diffusion*), spełnia warunki samostabilizacji, co w tym przypadku oznacza, że system zawsze znajdzie się w stanie ustalonym i jednostki lokalne nie muszą znać globalnego stanu systemu.

Algorytmy synchronizacji zegarów logicznych klasyfikuje się jako rozproszone, ponieważ ich podstawą jest wymiana wiadomości między procesami, bez wyróżnienia żadnego z nich.

Synchronizacja zegarów logicznych

Jako pierwszy w tej klasie opracowany został algorytm synchronizacji zegarów oparty na następstwie zdarzeń, opublikowany w 1978 r. przez Lamporta i do dziś najczęściej cytowany w literaturze [2, 5, 8]. Kolejność zdarzeń ocenia się na podstawie relacji „zdarzyło się poprzednio" (ang. *happened-before*), zapisywanej jako „→". Wyrażenie $a \rightarrow b$ oznacza „a wystąpiło przed b" i rozumiane jest, że zdarzenie a pojawiło się wcześniej niż b. Relacja „→" na zbiorze zdarzeń



Rys. 1. Uzyskanie bieżącej wartości czasu od aktywnego serwera

w systemie musi spełniać następujące dwa warunki:

1. Jeśli a i b są zdarzeniami w obrębie tego samego procesu i a wystąpiło przed b, to prawdziwe jest, że $a \rightarrow b$.
2. Jeśli a jest zdarzeniem wysłania wiadomości przez jeden proces i b jest zdarzeniem odbioru wiadomości przez inny proces, wtedy $a \rightarrow$.

Z warunków tych wynikają dalsze zależności, mianowicie:

- jeśli $a \rightarrow b$ i $b \rightarrow c$, wtedy $a \rightarrow c$,
- jeśli $a \rightarrow b$ i $b \rightarrow a$, to zdarzenia a i b są równoczesne.

Tak określona relacja następstwa zdarzeń jest prawdziwa tylko w procesach komunikujących się wzajemnie. Jeśli dwa zdarzenia x i y wystąpią w różnych, nie współpracujących ze sobą procesach, relacja $x \rightarrow y$ lub $y \rightarrow x$ nie jest słuszna, zdarzenia te są traktowane jako niezależne i nic nie można powiedzieć o kolejności ich wystąpienia, ponieważ nie ma to żadnego znaczenia dla funkcjonowania tych procesów.

Uszeregowanie zdarzeń jest możliwe dopiero po przypisaniu im czasu. Niech $C(a)$ oznacza czas wystąpienia zdarzenia a, natomiast $C(b)$ – czas wystąpienia zdarzenia b. Wartości czasu odniesione do poszczególnych zdarzeń muszą charakteryzować się tymi samymi właściwościami, co odpowiednie relacje. Jeżeli więc zdarzenie a wystąpiło przed $b (a \rightarrow b)$ to czas wystąpienia zdarzenia a musi mieć mniejszą wartość niż czas wystąpienia zdarzenia b:

$$\forall a, b: \text{jeśli} \rightarrow \text{to } C(a) < C(b)$$

Dodatkowo trzeba uwzględnić, że czas musi zawsze rosnąć i ewentualna korekta może być tylko wartością dodatnią, a nigdy ujemną.

Ustalenie relacji czasowej między zdarzeniami odbywa się zgodnie z zasadą „zdarzyło się przed". I tak każda wysłana wiadomość jest cechowana lokalnym czasem swojego nadawcy (przez wstawienie stempla czasu). Jeśli w miejscu odbioru czas ten będzie miał wartość mniejszą niż lokalny, zasada następstwa jest spełniona i zegar nie wymaga korekty. Jeśli natomiast czas niesiony przez wiadomość jest większy od lokalnego czasu węzła odbierającego, oznacza to, że zdarzenie wysłania ma wartość czasu większą niż wartość czasu związana ze zdarzeniem odbioru, co jest niezgodne z warunkiem 2. W takim wypadku zegar wymaga korekty. Proces odbierający ustawia czas na wartość większą niż otrzymana wartość stempla czasu. Zależności te ilustruje rysunek 2.

W chwili 24 węzeł wysyła wiadomość opatrzoną stemplem czasu (o wartości 24). Węzeł odbierający otrzymał tę wiadomość w swoim lokalnym czasie 20. W wyniku korekty czas ten ustawiony zostanie na wartości 25 i stanie się podstawą upływu dalszego czasu w tym węźle.

Opisany algorytm gwarantuje jednoznaczne uszeregowanie zdarzeń, ale przypisane im wartości czasu tylko w przybliżeniu określają moment ich wystąpienia. W systemach działających w czasie rzeczywistym czas pojawienia się zdarzenia jest w wielu przypadkach bardzo istotny. Dla tego rodzaju systemów zegary logiczne są niewystarczające i dlatego wymaga się kontroli czasu opartego na zegarze fizycznym.

Wybór procesu koordynującego

Wiele rozproszonych algorytmów opiera się na wykorzystaniu jednego wyróżnionego procesu w systemie, pełniącego funkcję koordynatora, generatora czasu, inicjatora pewnych aktywności itd. W odniesieniu do czasu może to np. wynikać z konieczności wyboru nowego procesu (węzła) pełniącego w systemie rolę *master*. Dla uproszczenia przyjmuje się, że w jednym węźle pracuje jeden proces, stąd pojęcie *węzeł* i *proces* używane są w dalszej części artykułu zamiennie.

W przypadku awarii węzła podającego czas, inne węzły dysponujące procesami o takich samych możliwościach muszą wybrać spośród siebie jeden proces, który zastąpi uszkodzony. Grupa algorytmów desygnujących jeden proces do pełnienia szczególnej funkcji nazywa się ogólnie algorytmami wyboru (elekcji). W algorytmach tych zakłada się, że każdy proces ma swój jednoznaczny, niepowtarzalny identyfikator, będący numerem skojarzonym najczęściej z adresem sieciowym. Zasadą algorytmów elekcji jest powołanie procesu o najwyższym numerze jako procesu koordynującego. Dodatkowym założeniem w algorytmach tego typu jest znajomość przez każdy węzeł numerów wszystkich innych procesów w systemie. Akcja wyboru zostaje podjęta w dowolnej chwili przez dowolny proces, który pierwszy stwierdzi brak procesu nadrzędnego w systemie. Jako przykłady algorytmów tej klasy zostaną przedstawione dwa: najwyższego numeru oraz pierścieniowy.

Algorytm najwyższego numeru

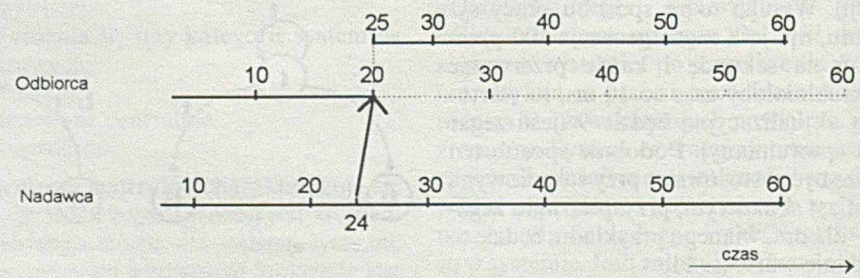
Proces, który stwierdził brak koordynatora, wysyła tzw. wiadomość wyborczą do wszystkich procesów o numerach większych niż jego identyfikator. Jeśli nie uzyska żadnej odpowiedzi, proces ten staje się koordynatorem, informując o tym wszystkie węzły przez przekazanie im odpowiedniej wiadomości (wiadomość typu koordynator).

Proces, który otrzymał wiadomość wyborczą, przyjmuje następujący tryb postępowania:

- odsyła do nadawcy potwierdzenie wiadomości wyborczej, informując go tym samym o swoim istnieniu,
- wysyła wiadomość wyborczą do wszystkich procesów o wyższych identyfikatorach niż jego numer,
- jeśli nie uzyska odpowiedzi od swoich adresatów, postępuje dalej jak proces, który wygrał wybory (koordynator).

Algorytm pierścienia

Wszystkie procesy w systemie tworzą pierścień logiczny, co oznacza, że każdy zna numer (i adres) swojego poprzednika i następcy. Proces, który wykryje, że nie funkcjonuje koordynator w systemie, buduje wiadomość wyborczą, zaopatrzoną we własny numer i wysyła ją do swojego następcy. Jeśli nie uzyska odpowiedzi, to uznaje węzeł za niepracujący i wysyła wiadomość do następnego węzła w pierścieniu, aż otrzyma potwierdzenie wiadomości. Zadaniem każdego kolejnego węzła, który otrzymał wiadomość, jest dopisanie do niej własnego identyfikatora oraz przesłanie jej do następcy. Po obiegu pierścienia wiadomość ta dotrze do inic-



Rys. 2. Aktualizacja czasu logicznego

jatora akcji, który rozpozna to po znalezieniu własnego numeru na liście. Tak otrzymana wiadomość zmienia charakter i staje się wiadomością typu koordynator, informującą o tym, kto przejmuje rolę procesu koordynującego w systemie oraz udostępnia uaktualnioną listę wszystkich procesów w systemie.

* * *

W rozproszonych systemach uwarunkowanych czasowo zagadnienie organizacji czasu nabiera szczególnego znaczenia, ponieważ stanowi podstawę synchronizacji procesów.

Pod pojęciem synchronizacji rozumie się współpracę między komunikującymi się procesami. Stosowane w systemach scentralizowanych metody i algorytmy synchronizacji nie mogą być zastosowane w rozproszonych systemach operacyjnych ze względu na fizyczne oddalenie zasobów od siebie, braku globalnej informacji stanu systemu oraz opóźnienia komunikacyjnego. Trudności w określeniu kolejności zdarzeń występujących w systemie wywołuje brak globalnego czasu systemowego. Stąd prace nad metodami synchronizacji procesów idą w dwóch kierunkach: sposobu ustalania globalnego czasu w systemie oraz uszeregowania zdarzeń bez takiej podstawy czasu, zastępując czas fizyczny czasem logicznym.

LITERATURA

[1] Ciuffoletti A.: Using simple diffusion to synchronize the clocks in distributed system. Proceedings of the 14th International Conference on Distributed Computing Systems, Poznań, June 21–24, 1994, IEEE Computer Society Press

[2] Gościński A.: Distributed Operating Systems. Addison-Wesley Publishing Company, 1991

[3] Harbour M.G., Klein M.H., Lehoczky J.P.: Timing Analysis for Fixed-Priority Scheduling of Real-Time Systems. IEEE Transactions on Software Engineering, Vol. 20, No. 1, Jan. 1994

[4] Lee I., Davidson S.B.: A Performance Analysis of Timed Synchronous Communication Primitivees. IEEE Transaction on Computers, Vol. 39, No. 9, Sept. 1990

[5] Levi S.T., Agrawala A.K.: Real Time System Design. McGraw-Hill Publishing Company, 1990

[6] Ramanathan P., Kandlur D.D., Shin K.G.: Hardware-Assisted Software Clock Synchronization for Homogeneous Distributed Systems. IEEE Transactions on Computers, Vol. 39, No. 4, April 1990

[7] Sha L., Rajkmur R., Lehoczky J.P.: Priority Inheritance Protocols: An Approach to Real-Time Synchronization. IEEE Transactions on Computers, Vol. 39, No. 9, Sept. 1990

[8] Tannenbaum A.S.: Modern Operating Systems. Prentice Hall, Englewood Cliffs, 1992.

POSTGRES rozszerzalny system zarządzania bazę danych

dokończenie ze s. 13

[4] Stonebraker M., Rowe L.: *The Design of Postgres*. In: Proc. ACM-SIGMOD Intern. Conf. Management of Data, Washington DC, May 1986

[5] Stonebraker M.: *Inclusion of New Types in Relational Data Base Systems*. In: Proc. of IEEE Intern. Conf. on Data Engineering, Los Angeles, 1986

[6] Stonebraker M., et al.: *Extensibility in Postgres*. IEEE Database Engineering, Sept. 1987

[7] Stonebraker M., et al.: *The Implementation of Postgres*. IEEE Transactions on Knowledge and Data Engineering, March 1990

[8] Stonebraker M., Kemnitz G.: *The Postgres next-generation database management system*. Communications of the ACM, 34 (10), pp. 78–92, 1991

Zamawiając prenumeratę INFORMATYKI wystarczy wypełnić poniższy blankiet, wyciąć go i dokonać wpłaty w banku lub na poczcie. Cena prenumeraty rocznej – 540 tys. zł.

ODCINEK DLA POCZTY (BANKU) ZŁ .....
SŁOWNIE ZŁOTYCH .....
BLANKIET WPŁAT DLA PRENUMERATORÓW
NAZWISKO .....
IMIĘ .....
ADRES .....
(ulica, nr domu i mieszkania)
(kod pocztowy) (miejscowość)
WYDAWNICTWO SIGMA-NOT sp. z o.o.
Zakład Kolportażu
00-950 Warszawa, skr. poczt. 1004
(Nazwa i siedziba posiadacza rachunku)
WPŁATA NA RACHUNEK NR 370015-1573-139-11
PAŃSTWOWY BANK KREDYTOWY III O/WARSZAWA
(Rachunek tylko dla prenumeraty czasopism)
DATOWNIK podpis przyjmującego
OPŁATA ZŁ .....
Prenumerata czasopism kolportowanych przez WYDAWNICTWO SIGMA-NOT sp. z o.o.
PRZEKAZ DLA WPŁAT NA RACHUNKI W BANKU

ODCINEK DLA POSIADACZA RACHUNKU ZŁ .....
SŁOWNIE ZŁOTYCH .....
NAZWISKO .....
IMIĘ .....
ADRES .....
(ulica, nr domu i mieszkania)
(kod pocztowy) (miejscowość)
WYDAWNICTWO SIGMA-NOT sp. z o.o.
Zakład Kolportażu
00-950 Warszawa, skr. poczt. 1004
(Nazwa i siedziba posiadacza rachunku)
WPŁATA NA RACHUNEK NR 370015-1573-139-11
PAŃSTWOWY BANK KREDYTOWY III O/WARSZAWA
(Rachunek tylko dla prenumeraty czasopism)
DATOWNIK podpis przyjmującego
OPŁATA ZŁ .....
Prenumerata czasopism kolportowanych przez WYDAWNICTWO SIGMA-NOT sp. z o.o.

ODCINEK DLA WPŁACAJĄCEGO ZŁ .....
SŁOWNIE ZŁOTYCH .....
BLANKIET WPŁAT DLA PRENUMERATORÓW
NAZWISKO .....
IMIĘ .....
ADRES .....
(ulica, nr domu i mieszkania)
(kod pocztowy) (miejscowość)
WYDAWNICTWO SIGMA-NOT sp. z o.o.
Zakład Kolportażu
00-950 Warszawa, skr. poczt. 1004
(Nazwa i siedziba posiadacza rachunku)
WPŁATA NA RACHUNEK NR 370015-1573-139-11
PAŃSTWOWY BANK KREDYTOWY III O/WARSZAWA
(Rachunek tylko dla prenumeraty czasopism)
DATOWNIK podpis przyjmującego
OPŁATA ZŁ .....
Prenumerata czasopism kolportowanych przez WYDAWNICTWO SIGMA-NOT sp. z o.o.
WPŁACAJĄCY WYPEŁNIA RÓWNIEZ NA ODWROCIE

Największe firmy informatyczne w 1993 roku

Wśród list rankingowych światowych firm informatycznych największą popularnością cieszy się publikacja ogłaszana w czerwcu każdego roku przez czasopismo Datamation. INFORMATYKA już w przeszłości (nr 2–3 z 1982 r.) udostępniała swym Czytelnikom te dane. Tegoroczna publikacja, poza standardowymi listami „pierwszych 100" firm w Ameryce Płn. i na świecie, przynosi wyniki ankiety na temat kryteriów wyboru dostawców techniki informatycznej przez użytkowników oraz 10 głównych firm w dziewięciu wybranych dziedzinach. Przedstawiamy naszym Czytelnikom główne fragmenty tego raportu, ograniczając się na ogół do pierwszych części publikowanych tabel, niekiedy uzupełniając je własną statystyką (np. zestawienie owych dziewięciu dziedzin).

Wspominając swe pierwsze raporty, autorzy publikacji podkreślają, że wówczas brak było standaryzacji i odbiorca musiał bardzo starannie dobierać dostawcę, gdyż zły wybór mógł prowadzić do ogromnych strat ze względu na niewymienność sprzętu oraz bezużyteczność posiadanego oprogramowania. Dziś nie ma już takiego niebezpieczeństwa, ale problem staje się bardziej złożony i należy więcej wiedzieć o wyrobach, które powinny być obsługiwane przez wielu dostawców.

Dlatego zapytano ponad 300 użytkowników, czym kierują się przy wyborze dostawcy, a następnie, jakie firmy spełniają te wymagania. Okazało się, że w dziesięciopunktowej skali stawiają oni na pierwszym miejscu jakość i niezawodność wyrobu (9,6 pkt.), następnie jego sprawność (9,2), jakość oferowanych usług (8,8), a dopiero później wiarygodność dostawcy (8,2), stosunek wydajności do ceny (8,1), łatwość zawierania transmisji z dostawcą (7,7), spełnianie przez dostawcę standardów branżowych (7,6), perspektywiczność (7,4) i wreszcie na końcu stabilność finansową dostawcy (7,0). Stosując te kryteria do 31 firm amerykańskich uzyskano zestawienie, z którego przytaczamy 10 pierwszych pozycji (tab. 1), gdzie komb. – oznacza współczynnik będący kombinacją wszystkich pozostałych.

Ostatnie miejsce w pełnej tabeli zajmuje Computervision z kombinowaną punktacją 5,2, a poprzedzają ją Wang i Tandem (po 5,4).

Firmy o wysokim współczynniku kombinowanym zajmowały wysokie pozycje na listach rankingowych i wykazywały tendencje do poprawiania tych pozycji.

Tabela 2 przedstawia 50 czołowych światowych firm informatycznych zestawionych według przychodów z technologii informatycznej w 1993 r. w mln dolarów USA. Podano tu również przychody w 1992 r. oraz procentową zmianę tych wielkości.

Widzimy, że IBM utrzymał swą pierwszą pozycję i mimo spadku przychodów

W przypadku uprawnień do prenumeraty ulgowej prosimy wszystkie odcinki przekazu ostemplować pieczęcią szkoły, uczelni lub stowarzyszenia naukowo-technicznego. Cena prenumeraty rocznej ulgowej – 378 tys. zł.

Celem dokonania wpłaty należy wypełnić drukowanymi literami wszystkie części blankietu i złożyć blankiet wraz z gotówką w Urzędzie Pocztowym, oddziale PKO lub Banku

Tytuł czasopisma	Symb. czas.	Liczba egz.	Wartość
INFORMATYKA			
RAZEM:			
OKRES PRENUMERATY:			

Tytuł czasopisma	Symb. czas.	Liczba egz.	Wartość
INFORMATYKA			
RAZEM:			
OKRES PRENUMERATY:			

Tytuł czasopisma	Symb. czas.	Liczba egz.	Wartość
INFORMATYKA			
RAZEM:			
OKRES PRENUMERATY:			

UWAGA: Prenumerata ulgowa będzie przyjęta tylko w przypadku gdy **wszystkie odcinki** blankietu będą poświadczone pieczęcią SNT lub szkoły prenumeratora uprawnionego do zniżki.

Tabela 1

Firma	Wydajność wyrobu	Jakość i niezgodność wyrobu	Jakość usług	Wiarygodność dostawcy	Zgodność ze standardami	Stosunek wydajności do ceny	Stabilność finansowa	Łatwość współpracy z dostawcą	Perspektywy	Komb.
Microsoft	7,8	7,5	6,9	7	7,2	7,7	8,4	7,4	7,3	7,5
H-P	7,5	7,8	7,2	7,5	7,3	6,8	7,7	7,0	7,1	7,3
Compaq	7,8	8,0	7,0	7,8	7,4	6,9	7,1	6,8	6,8	7,3
Sun	7,8	7,8	7,1	7,2	7,0	6,8	7,1	6,7	6,8	7,1
Novell	7,3	7,3	6,8	7,1	6,9	6,5	7,6	6,6	7,0	7,0
IBM	7,4	7,7	7,4	7,4	6,7	6,0	7,2	6,7	6,5	7,0
AT & T	6,9	7,0	6,6	7,1	6,9	6,1	7,7	6,6	6,5	6,8
Dell	6,8	7,0	6,8	7,0	7,1	7,1	6,0	7,1	6,6	6,8
Digital	7,0	7,1	6,9	7,0	6,7	6,4	6,4	6,6	6,7	6,8
Seagate	6,9	6,8	6,2	6,7	7,3	7,0	6,4	6,6	6,6	6,7

o 2,8% są one prawie trzykrotnie wyższe od następnej na liście Fujitsu. Tak więc kolejność pierwszej trójki nie zmieniła się i po IBM idą dwie firmy japońskie, które zwiększyły w 1993 r. swe przychody o ponad 8%. Pierwsza zmiana nastąpiła na czwartej pozycji, gdzie dynamicznie rozwijający się Hewlett-Packard (wzrost przychodów o 23%) wyprzedził nie mogącego się wciąż przystosować do nowych warunków Digitala (spadek przychodów o 3,7%). Dalsze dwie pozycje w tabeli nie uległy zmianie, przy czym Hitachi wykazała najwyższy wzrost przychodów wśród dotychczas wymienionych firm japońskich (o 11,2%), a AT&T – największy dotychczas notowany spadek (o 5,5%). Na ósmym miejscu znalazła się Toshiba awansując w tabeli o dwa miejsca dzięki wzrostowi przychodów o 18,4%. Na dziewiątym miejscu znalazła się amerykańska firma EDS (Electronic Data System Corp. z Plano w Teksasie) awansując aż o 6 miejsc w tabeli przy wzroście przychodów zaledwie o 3,5%, podczas gdy Apple przesunęła się w górę tylko o jedno miejsce przy wzroście dochodów o ponad 10%. Spadły natomiast firmy zachodnioeuropejskie jak Siemens, Olivetti, Bull, ICL i Cap Gemini o 2–8 miejsc w tabeli, przy spadku przychodów o 10–14%, oraz niektóre japońskie (Matsushita i Oki), przy mniejszym zakresie spadku. Z firm amerykańskich znaczniejsze tendencje spadkowe wykazały Unisys (o trzy miejsca i 8,1%), Amdahl (o 18 miejsc i 33,4%), a z drugiej pięćdziesiątki Maxtor (o 8 miejsc i 11,4%) i Intergraph (o 6 miejsc i 10,8%). Uwzględniając pełną, 100-pozycyjną listę, tabela 3 przedstawia 10 firm o najwyższym procentowym wzroście przychodów z technologii informatycznej.

Znalazły się tu przede wszystkim małe firmy amerykańskie, z których trzy pierwsze po raz pierwszy zjawiły się w 1993 r. na tej liście, a jedynie Compaq znajduje się w pierwszej piętnastce.

Tabela 2

Miejsce w 1993 r.	Miejsce w 1992 r.	Firma	Przychody w 1993 r.	Przychody w 1992 r.	Zmiana procentowa
1	1	IBM	62716,0	64520,0	−2,8
2	2	Fujitsu	21871,9	20142,2	8,6
3	3	NEC	16674,8	15395,0	8,3
4	5	HP	15600,0	12688,0	23,0
5	4	Digital	13637,0	14162,0	−3,7
6	6	Hitachi	12629,1	11352,0	11,2
7	7	AT & T	9860,0	10450,0	−5,6
8	10	Toshiba	8819,7	7448,7	18,4
9	15	EDS	8507,3	8218,9	3,5
10	11	Apple	7900,0	7173,7	10,1
11	8	Siemens Nixdorf	7225,5	8345,1	−13,4
12	9	Unisys	7200,5	7832,0	−8,1
13	18	Compaq	7200,0	4100,0	75,6
14	12	Olivetti	5070,2	5762,0	−12,0
15	14	Matsushita	5050,7	5060,8	−0,2
16	16	Canon	5033,0	4633,6	8,6
17	13	Groupe Bull	5000,0	5715,1	−12,5
18	19	Sun Microsystem	4493,0	3832,0	17,2
19	20	Microsoft	4110,0	3253,0	26,3
20	17	ICL	3915,7	4354,8	−10,1
21	21	NTT	3905,6	3119,5	25,2
22	23	Xerox	3330,0	3016,0	10,4
23	22	Seagate	3114,0	3079,4	1,1
24	26	Mitsubishi	3042,5	2589,4	17,5
25	24	Andersen Consulting	2876,3	2720,0	5,7
26	36	Dell	2870,0	2013,9	42,5
27	25	Nihon Unisys	2697,8	2646,1	2,0
28	28	CSC	2502,0	2474,4	1,1
29	34	ADP	2339,1	2075,0	12,7
30	33	Ricoh	2290,0	2124,6	7,8
31	37	TRW	2251,0	1855,0	21,3
32	29	Northern Telecom	2250,0	2100,0	7,1
33	30	Oki	2245,6	2289,9	−1,9
34	32	Conner Peripherals	2200,0	2240,0	−1,8
35	38	CA	2054,8	1770,8	16,0
36	35	Tandem	2023,0	2066,3	−2,1
37	40	Quantum	2002,0	1536,2	30,3
38	52	AST Research	1971,0	1140,5	72,8
39	31	Cap Gemini	1946,9	2252,9	−13,6
40	48	Acer	1898,0	1192,2	59,2
41	47	Intel	1844,2	1230,0	49,9
42	39	Seiko Epson	1768,6	1649,8	7,2
43	54	Gateway 2000	1731,7	1107,1	56,4
44	46	Oracle	1692,9	1310,0	29,2
45	27	Amdahl	1680,5	2524,7	−33,4
46	49	First Data	1500,0	1205,0	24,5
47	53	Samsung	1494,4	1117,7	33,7
48	41	StorageTek	1405,0	1521,5	−7,7
49	45	Lexmark	1350,0	1350,0	0
50	59	Western Digital	1267,0	943,4	34,4

Tabela 3

Punktacja	Firma	Procentowy wzrost wpływów
1	EMC	124,2
2	Cisco	96,1
3	SynOptics	81,2
4	Compaq	75,6
5	AST Research	72,8
6	Science Application	63,2
7	Acer	59,2
8	Martin Marietta	58,5
9	Gateway 2000	56,4
10	Intel	49,9

Wróćmy do tabeli 2. Pojawiło się w niej osiem nowych firm – wspominane trzy z tabeli 3 (odpowiednio na pozycjach 63, 75 i 81) oraz pięć firm na pozycjach poniżej 93. Firma Science Application przesunęła się o 27 pozycji w górę tabeli, Martin Marietta o 25 pozycji, 3 Com o 15 i AST Research o 14. Wśród firm, które pozostały w pierwszej setce, ale znacznie zmniejszyły swoje przychody w stosunku do 1992 r., należy wymienić Wang (spadek o 39,3% i 23 pozycje) i wspomnianą Amdahl oraz Memorex Telex (odpowiednio o 24% i 18 pozycji). Ponadto nastąpiły niewielkie zmiany, a pięć firm w pierwszej siódemce nie zmieniło swych pozycji.

Tabela 4 podaje 30 czołowych firm północnoamerykańskich. Znajduje się w niej nieco więcej danych niż w tabeli 2, m.in. jaki procent przychodów firmy stanowi technologia informatyczna, a także dane o zysku w latach 1992–1993.

Te dodatkowe dane pokazują, że 39% spośród czołowych firm poniosło w 1993 r. straty (w porównaniu z 14% firm w 1992 r.), a 43% wykazało pogorszenie zysku lub zwiększenie strat. Nawet firmy zwiększające znacznie swoje przychody i awansujące na liście, jak Dell czy AST Research, przyniosły w 1993 r. straty. W wartościach bezwzględnych największe straty poniosła IBM (ponad 8 mld dolarów), choć w ostatnich kwartałach i tu wystąpiła poprawa. Procentowo największą niekorzystną zmianę odnotował Amdahl (ponad 72 razy). Największy zysk bezwzględny osiągnął Intel (prawie 2,3 mld dolarów), a procentowo – w podanej części tabeli – Xerox (ponad 160%). Wśród pełnej listy amerykańskiej odnotowano także ponad pięciokrotne zwiększenie zysku (Cray) i ponad trzykrotne (EMC).

W cytowanej części tabeli 70% firm, to firmy czysto informatyczne. Najniższy udział tej techniki widzimy w AT & T (14,7%) – w pełnej tabeli tylko 1,6% notowane jest u Boeinga. Zmiany w tej tabeli podobne są do zmian w tabeli 2, jeśli uwzględnimy wyeliminowanie firm z innych kontynentów.

Przejdźmy teraz do poszczególnych dziedzin rozwoju informatyki. Nie będziemy tu przytaczać tabel dla wszystkich dziewięciu wybranych dziedzin. Natomiast sumując przychody pierwszych dziesięciu firm w danej dziedzinie uzyskamy względną ocenę wagi danej dziedziny w 1993 r. Określono także procentowe zmiany tych przychodów w porównaniu do roku poprzedniego. Pokazuje to tabela 5. Sumy przychodów są wyrażone w mln dolarów USA.

Jak widać, największy procentowy przyrost wykazały komputery osobiste (36,3%), a następnie komunikacja (17,2%) i usługi (13,9%). Zmniejszenie natomiast wykazały duże systemy (37,6%), oraz w niewielkim stopniu stacje robocze (3,6%) i urządzenia peryferyjne (2,5%).

Przyczyną spadku sprzedaży dużych systemów jest odwrócenie proporcji między zapotrzebowaniem na moc oblicze-

Tabela 4

Miejsce w 1993 r.	Miejsce w 1992 r.	Firma	Przychody w 1993 r.	Przychody w 1992 r.	Zmiana procentowa	Procent technologii informatycznej	Zysk w 1993 r.	Zysk w 1992 r.	Zmiana procentowa
1	1	IBM	62716,0	64520,0	−2,8	100,0	−8101,0	−6870,0	−17,9
2	3	HP	15600,0	12688,0	23	76,8	1200,0	549,0	118,6
3	2	Digital	13637,0	14162,0	−3,7	100,0	−72,0	−2501,0	97,1
4	4	AT & T	9860,0	10450,0	−5,6	14,7	−3974,0	3807,0	−204,4
5	7	EDS	8507,3	4870,0	74,7	99,4	724,0	635,5	13,9
6	6	Apple	7900,0	7173,3	10,1	100,0	86,0	525,7	−83,6
7	5	Unisys	7200,5	7832,0	−8,1	93,0	565,4	361,0	56,6
8	8	Compaq	7200,0	4100,0	75,6	100,0	462,0	213,2	116,7
9	9	Sun	4493,0	3832,0	17,2	100,0	188,2	127,9	47,2
10	10	Microsoft	4110,0	3253,0	26,3	100,0	1036,0	833,0	24,4
11	12	Xerox	3330,0	3016,0	10,4	22,8	620,0	−1020,0	160,8
12	11	Seagate	3114,0	3079,4	1,1	100,0	151,0	222,9	−32,3
13	13	Andersen	2876,3	2720,0	5,7	100,0	brak danych		
14	20	Dell	2870,0	2013,9	42,5	100,0	−35,8	101,6	−135,2
15	15	CSC	2502,0	2474,4	1,1	100,0	89,1	75,7	17,7
16	18	ADP	2339,1	2075,0	12,7	100,0	310,8	272,0	14,3
17	21	TRW	2251,0	1855,0	21,3	28,3	195,0	194,0	0,5
18	16	Northern Telecom.	2250,0	2100,0	7,1	27,6	−878,0	536,0	−263,8
19	17	Conner Periph.	2200,0	2240,0	−1,8	100,0	−445,0	121,0	−467,8
20	22	CA	2054,8	1770,8	16	100,0	341,6	204,2	67,3
21	19	Tandem	2023,0	2066,3	−2,1	100,0	−510,0	70,9	−818,9
22	23	Quantum	2002,0	1536,2	30,3	100,0	−9,5	84,8	−111,2
23	32	AST Research	1971,0	1140,5	72,8	100,0	−50,0	57,6	−186,8
24	29	Intel	1844,2	1230,0	49,9	21,0	2295,0	1066,5	115,2
25	33	Gateway 2000	1731,7	1107,1	56,4	100,0	100,1	70,0	43,0
26	28	Oracle	1692,9	1310,0	29,2	100,0	197,8	89,0	122,2
27	14	Amdahl	1680,5	2524,7	−33,4	100,0	−580,0	−7,0	−8215,4
28	30	First Data	1500,0	1205,0	24,5	100,0	173,0	141,0	22,7
29	24	StorageTek	1405,0	1521,5	−7,7	100,0	−78,0	15,5	−603,2
30	27	Lexmark	1350,0	1350,0	0	75,0	brak danych		

niową i stosunkiem wydajności do ceny. W rezultacie IMB zmniejszyła sprzedaż dużych maszyn o 47%, lecz nadal jej udział w tej dziedzinie wynosi 53%. Podobnie (o 42%) wystąpił tu spadek u Amdahla i w mniejszym stopniu (o 16%) u Unisys. Rozwój w dziedzinie dużych systemów odnotował Intel (54%), oraz Cray, HP i Digital (odpowiednio 18, 11 i 10%). Obserwuje się tu zastępowanie dużych maszyn komputerami Unixowymi. Cena za milion operacji na sekundę spadła w tej dziedzinie z 45 do 30 tys. dolarów, a niektóre przewidywania mówią o 4 tys. dolarów w 1996 r. Taki rozwój sytuacji pociągnął za sobą redukcję zatrudnienia w ośrodkach wytwarzających duże systemy i zwrot ku tańszym technologiom. Pierwsze wyniki za 1994 r. mówią, że dało to rezultaty i dalszy spadek został zahamowany.

W dziedzinie średnich systemów największe sukcesy odnotował HP (58% wzrostu przychodów i osiągnięcie 16% udziału rynku). Większy procent wzrostu sprzedaży odnotowała tylko Data General, lecz jej udział wynosi tylko 4%. IBM zachowało pierwszą pozycję z udziałem 43%, ale jego przychody w tej dziedzinie nie zmieniły się od poprzedniego roku. Spadek natomiast wykazał Digital (o 20% przy 15% udziale) oraz Stratus (odpowiednio 19% i 3%). Największy rozwój wykazały tu systemy Unixowe, stosowane często do zarządzania bazami danych.

W dziedzinie oprogramowania tylko IBM wykazało niewielki (4%) spadek przychodów pozostając jednak na pierwszym miejscu z 36% udziałem. Najlepiej wypadł Microsoft (26% wzrost sprzedaży i 12% udziału) oraz Oracle (29% i 4%) i WordPerfect (22% i 2%). Interesujące, że Borland znalazł się poza pierwszą dziesiątką. Wzrósł tu udział systemów klient-serwer, a takżej pojawiły się pierwsze systemy ukierunkowane na obiekty, których udział będzie zwiększał się w następnych latach. Obserwuje się tu nadal koncentrację dużych firm, jak np. zakupienie przez Novella firm WordPerfect i Quattro Pro.

Tabela 5

Dziedzina	Przychody pierwszych 10 firm	Zmiana (%)
Usługi	33 750	13,9
Komputery osobiste	33 400	36,3
Urządzenia peryferyjne	29 430	−2,5
Oprogramowanie	22 940	7,3
Utrzymywanie systemów	20 290	1,3
Średnie systemy	12 960	6,3
Komunikacja	11 860	17,2
Stacje robocze	9040	−3,6
Duże maszyny	7900	−37,6

W dziedzinie o największych dochodach, jaką są usługi, nie odnotowano w 1993 r. spadku przychodów w pierwszej dziesiątce firm. Największy przyrost wykazało IBM (32% przy udziale 21%), a następnie First Data (24% i 3%) oraz Digital i Unisys (po 19% przy udziałach odpowiednio 4 i 3%). Mająca 18% udziału firma EDS wykazała 4% wzrostu, natomiast najmniejszy (1% przy udziale 5%) przyrost przychodów uzyskała firma CSC. Również tu widać rozwój technologii klient-serwer, a firmy, które potrafiły zaoferować tu tańsze rozwiązania uzyskały większy wzrost.

Komputery osobiste wykazały w 1993 r. największy procentowy przyrost sprzedaży. Tu również wzrost odnotowała cała pierwsza dziesiątka. Najwięcej (76% przy udziale 21%) uzyskał Compaq, który znalazł się na drugim miejscu tabeli, a za nim ustępując nieco – AST Research (72% i 6%) oraz Gateway 2000 (56% i 5%). Utrzymujący nadal pierwszą pozycję IBM wykazał wzrost o 23% przy udziale 29%, natomiast najmniejszy rozwój wykazały AT & T (2% i 3%) oraz Apple (9% i 17%). Wśród najnowszych wyrobów w tej dziedzinie widzimy wiele komputerów typu notebook, jak np. Compaq Concerto, IBM ThinkPad i AST Research Bravo, natomiast komputery z mikroprocesorem PowerPC pojawią się masowo dopiero w 1994 r.

Komunikacja danych zajęła drugie miejsce wśród najszybciej rozwijających się dziedzin, chociaż mający największy udział w rynku (23%) IBM nie odnotował tu wzrostu przychodów. Prawie dwukrotny wzrost (96%) wykazała firma Cisco z udziałem 7%, a następnie SynOptics (81% i 5% udziału) oraz Cabletron (49% i 4%). Duże firmy telekomunikacyjne, jak AT & T oraz Northern Telecom, mające po 16–19% udziału w rynku, wykazały wzrost tylko 5–7%. Można tu mówić o pierwszych transakcjach dotyczących tzw. autostrad informatycznych, lecz większy rozwój przewiduje się tu dopiero w następnych latach.

Dziedzina stacji roboczych zajęła przedostatnie miejsce (przed wielkimi maszynami), zarówno jeśli chodzi o wartość przychodów, jak i współczynnik rozwoju (spadek o 3,6% w porównaniu z 1992 r.). Tylko HP, zajmujący z udziałem 22% drugie miejsce w tej branży, uzyskał wzrost przychodów o 55%. Wzrost o 28% wykazała charakteryzująca się udziałem 9% firma Silicon Graphics, a o 17% – lider stacji roboczych – Sun, wykazujący udział 26%. Pozostałe firmy pierwszej dziesiątki miały bądź nieznaczny wzrost (Wyse o 4%), bądź zmniejszenie przychodów. Rekordzistką okazała się tu firma Intergraph, w której zmniejszenie wyniosło aż 66% (udział 2%), a po niej Computervision (odpowiednio 40% i 1%) oraz Unisys (31% i 4%). Również IBM, mający udział 13% rynku odnotowała zmniejszenie o 25%. Tańsze stacje robocze są wypierane przez bardziej rozbudowane systemy komputerów osobistych, które stają się konkurencyjne pod względem ceny i parametrów. Rozwijały się natomiast stacje bardziej złożone, dla których uzyskano odpowiednio wysoki stosunek wydajności do ceny.

Jeśli chodzi o urządzenia peryferyjne, to ta trzecia pod względem przychodów dziedzina wykazała w sumie niewielki spadek w porównaniu z poprzednim rokiem, a to z przyczyny spadku przychodów jej lidera – IBM, który mając udział 15% odnotował w 1993 r. spadek 27%. Podobnie było u Digitala (spadek 22 przy udziale 6%), oraz niewielki spadek (2% przy udziale 6%) w firmie Conner Peripherals. Pozostałe firmy pierwszej dziesiątki odnotowały wzrost przychodów – największy firm Lexmark i Quantum (odpowiednio 34 i 30% przy udziale 5% każdej z firm), a HP i Xerox po 13% (odpowiednio przy udziale 13 i 8%). Szybko rozwijającymi się obszarami są tu ulepszone systemy dyskowe, w tym dyski optyczne, stające się powoli standardem w komputerach osobistych.

Utrzymywanie systemów znalazło się pośrodku wyodrębnionych dziedzin z niewielkim (1,3%) współczynnikiem wzrostu przychodów. Liderem jest tu również IBM, który przy udziale 33% wykazał 4% spadek. Większy spadek wykazały firmy Unisys i Memory Telex (odpowiednio 14 i 10% przy udziale 2–3%). Natomiast firma Sun wykazała wzrost 15% przy udziale 2%, a HP – wzrost 5% przy udziale 13%. Przewiduje się, że dochody z tej dziedziny w najbliższych latach będą zmniejszać się, gdyż nowe systemy wymagają mniejszych nakładów na ich utrzymanie. Występuje tu też zjawisko obsługi systemów jednych firm przez inne. Natomiast część usług, oferowanych dotychczas za darmo, została zaliczona do płatnych.

Uwzględniając pierwsze trzy czołowe firmy we wszystkich wymienionych dziedzinach, w siedmiu z nich na pierwszym miejscu znajduje się IBM, a w dwu pozostałych – na trzecim. HP pojawia się tu czterokrotnie – trzy razy na drugim miejscu i raz na trzecim, a Digital dwukrotnie – raz na drugim a raz na trzecim. Pozostałych 12 firm występuje tylko jeden raz.

dokończenie na s. 28

Osobiste stacje robocze

Dość powszechnie uznawany jest dziś podział komputerów klasy desktop na osobiste i stacje robocze. Te pierwsze realizują standard IBM PC, pracujący pod kontrolą systemu operacyjnego **DOS/Windows**, stosunkowo mało wydajny pod względem zdolności obliczeniowych, za to bardzo rozpowszechniony i bardzo dobrze oprogramowany pod kątem potrzeb biurowych (edytory, arkusze kalkulacyjne, użytkowe bazy danych). Pod pojęciem stacji roboczej natomiast rozumie się komputer o wielokrotnie większej niż PC zdolności obliczeniowej, przystosowany do realizowania złożonych zadań obliczeniowych i dużych projektów inżynierskich, z procesorem klasy **RISC**, pracujący pod kontrolą którejś z mutacji systemu **Unix**, za to w mniejszym niż PC stopniu przygotowany do prac biurowych. Świat PC osiągnął stan wysokiego nasycenia wydajnym i tanim oprogramowaniem. Tymczasem różnice między platformami **RISC – owymi** i środowiskami poszczególnych Unixów spowodowały, że analogiczny proces upowszechnienia ominął świat stacji roboczych.

Rozróżnienie **"PC – stacja robocza"** powoduje, że w pracowniach inżynierskich i naukowych przeważnie muszą stać obok siebie dwa komputery: stacja robocza do prac "poważnych" i PC, za pomocą którego realizowane są zadania związane z organizacją pracy, korespondencją, opracowywaniem publikacji, itp. Kluczem do rozwiązania tego problemu byłoby jednorodne, skalowalne środowisko hardware'u i software'u, które byłoby w stanie sprostać wymaganiom zaawansowanych aplikacji.

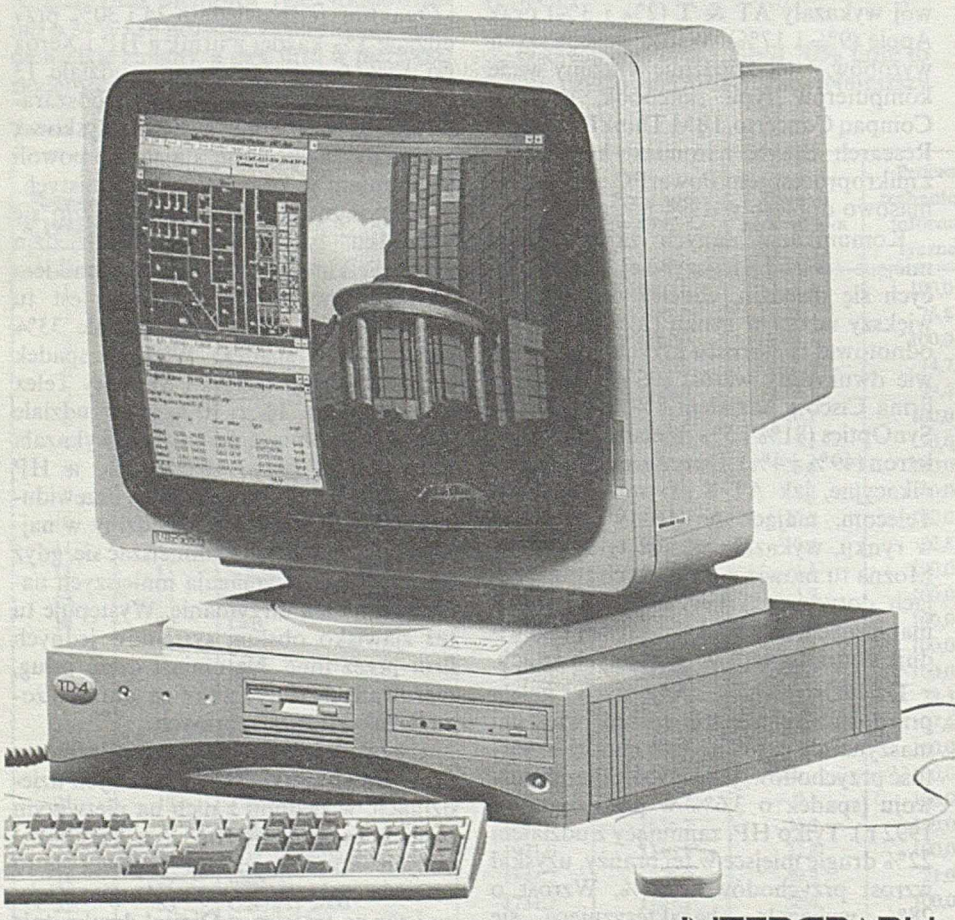
Filozofię, wypracowaną przez firmę Intergraph, reprezentuje pojęcie **"osobistej stacji roboczej"** (personal workstation). Zakłada ona, że skuteczne rozwiązanie zarysowanego powyżej dylematu oferuje połączenie nowego, wielofunkcyjnego systemu operacyjnego **Microsoft Windows NT** z nową generacją wysokowydajnych komputerów opartych na procesorach **Pentium**. Dalszy rozwój procesora Pentium oraz systemu Windows NT w kierunku systemów wieloprocesorowych obiecuje bardzo szybki postęp już w najbliższej przyszłości.

Osobiste stacje robocze **Intergraphu**, tworzące obecnie rodzinę **TD (Technical Dekstop)**, są zbudowane w oparciu o następujące założenia konstrukcyjne:

- ❑ intelowskie procesory i486 (low end) i Pentium (high end) z zegarem 66, 90 i 100 MHz, 512 kB cache, 256 MB RAM
- ❑ architektura symetrycznej wieloprocesorowości (dwa procesory Pentium w stacjach TD-4 i TD-5)
- ❑ wbudowany kontroler Ethernetu
- ❑ dwa łącza Fast SCSI-2
- ❑ możliwość wbudowania napędów CD-ROM
- ❑ szybka magistrala wewnętrzna typu EISA lub PCI
- ❑ akceleratory grafiki dwymiarowej G90 i G91 z 4 MB VRAM
- ❑ akceleratory grafiki trójwymiarowej: GLZ (24 MB VRAM) i GLI (34 MB VRAM) z wbudowanym sprzętowo cieniowaniem Gourouda, podniesioną rozdzielczością, renderingiem obrazów złożonych, antyaliasingiem, 10 – bitową korekcją gamma i gotowością do tworzenia obrazów stereoskopowych. Podsystem GLI posiada dodatkowo bogate możliwości operowania teksturą.

Obecnie rodzina **Technical Desktop** składa się z pięciu członków:

- ❑ **TD-1**: model ekonomiczny, dorównujący stacjom roboczym pod względem wydajności grafiki i przepustowości kanałów wejścia/wyjścia. Główne elementy: procesor 486, szyna lokalna VLB, kontroler graficzny S3.
- ❑ **TD-2**: Pentium 66 MHz, max. 192 MB RA, zwiększona pamięć cache, procesor graficzny G90, szyna lokalna PCI.
- ❑ **TD-3**: Pentium 90 MHz, szyna lokalna PCI, Fast SCSI-2, procesor graficzny G90 lub G91,
- ❑ **TD-4**: 2xPentium 90 MHz, max. 256 MB RAM, sterownik graficzny G91 lub GLZ, możliwość pracy z dwoma monitorami jednocześnie, dysk twardy SCSI 1 GB.
- ❑ **TD-5**: Rozszerzenie modelu TD-4 m.in. o możliwość pracy ze sterownikiem graficznym GLI, model najbardziej przygotowany do rozbudowy.



INTERGRAPH

Stacje Technical Desktop współpracują z wysokiej klasy monitorami **InterVue** z ekranami o średnicy 20, 21 lub 27 cali, które zdolne są wyświetlić do 2 mln punktów obrazowych przy częstości odświeżania obrazu 150 Hz. Parametry te, jak również pokrycie warstwą antyodblaskową, automatyczne centrowanie i wymiarowanie obrazu, gwarantują stabilny, pozbawiony migotań obraz o bardzo wysokiej jakości.

Największy na świecie ośrodek rozwoju oprogramowania dla Windows NT, **Intergraph** zdecydował się przenieść większość swoich aplikacji CAD – owskich do tego właśnie środowiska. Znaczna część tych aplikacji jest już dziś dostępna na platformie NT.

O/7/94

Prosto z firm ● Prosto z firm

Zysk American Power Conversion (APC) – znanego producenta zasilaczy awaryjnych UPS – wzrósł w II kw. br. o 51% w stosunku do analogicznego okresu roku ubiegłego i wyniósł 15,5 mln USD, zaś przez całe pierwsze półrocze – 29,2 mln USD. W tym okresie sprzedaż osiągnęła 160,9 mln USD – wzrost o 53% w stosunku do analogicznego okresu 1993 r. Sukcesy te, jak stwierdził prezes firmy, wynikają z coraz większego zainteresowania użytkowników sprawną pracą komputerów i sieci. Przyszłość APC też zapowiada się nie najgorzej. Microsoft wybrał APC Smart-UPS do ochrony opracowywanego obecnie serwera wideo na żądanie. Ogromny potencjalny rynek tworzą prywatni użytkownicy komputerów, którym coraz bardziej zależy na bezawaryjnej pracy komputera; obecnie tylko co 18 PC (w Ameryce) wyposażony jest w UPS.

* * *

Obroty Domu Handlowego Informatyki/Computer 2000 za pierwsze półrocze br. wyniosły 336,5 mld zł (ok. 15,16 mln USD) – wzrost o ponad 100% w stosunku do ub. r. Jest to przede wszystkim efekt poszerzenia oferty o sprzęt Compaqa, produkty SCO, Informixa, kas fiskalnych IBM, urządzeń awaryjnych firm BEST i Merlin. DHI toczy zawziętą walkę z nielegalnym kopiowaniem programów, a jednym ze sposobów jest organizowanie na giełdach komputerowych punktów sprzedaży legalnych programów po promocyjnych cenach.

* * *

Grupa Acer, założona w 1976 r., produkująca PC, multimedia, notebooki i urządzenia peryferyjne, posiadająca ok. 70 oddziałów w 27 krajach i zatrudniająca ponad 8 tys. pracowników, awansowała z 48. na 40. miejsce wśród największych firm przemysłu komputerowego na świecie. W I półroczu br. sprzedaż grupy osiągnęła 1,36 mld USD – wzrost o 76% w stosunku do analogicznego okresu ub. r. W 1991 r. Acer ustanowił standard przemysłowy, opracowując technologię szybkiego upgrade'u procesora zwaną ChipUp. W 1993 r. wprowadził na rynek komputery PC z 64-bitowym procesorem RISC. Dystrybutorem produktów Acera w Polsce jest DHI.

* * *

Klienci LIM Centre przy hotelu Marriott w Warszawie mają bezpośredni dostęp do obejmującej 30 krajów, na wszystkich kontynentach, światowej sieci komputerowej INTERNET. Usługę tę oferuje PL-net – polski oddział EUnet – europejskiej sieci komputerowej. Dzięki niej można z własnego komputera wysyłać pocztę elektroniczną, brać udział w międzynarodowej konferencji komputerowej NEWS oraz korzystać z usług sieci Internet. Internet dla LIM Centre powstał w ramach rozpoczęcia działalności polskiego EUnet. Usługi EUnet rozliczane są w ramach usług telekomunikacyjnych LIM.

* * *

NEC Home Electronics Ltd. Sharp Elektronics Corporation i Zenith Data Systems zapowiedziały zbudowanie przenośnego sprzętu opartego na przenośnym asystencie cyfrowym WinPad – programowej architekturze Microsoft At Work. Poza tym Syware Inc. i Navigation Technologies Inc. zamierzają opracować aplikacje dla tego urządzenia. Oddział Microsoft WGA opracowuje oprogramowanie umożliwiające łączenie się z urządzeń WinPad z serwerami poczty elektronicznej Microsoft Mail. Pierwsze egzemplarze asystenta cyfrowego WinPad i oprogramowanie do niego mają pojawić się pod koniec br.

Prosto z firm ● Prosto z firm

Zastosowanie sieci neuronowych...

dokończenie ze s. 6

Autorom projektu udało się zrealizować narzędzie umożliwiające identyfikację procesów niesterowalnych. Dostarcza ono informacji o różnicy między identyfikowaną klasą procesów a klasą pożądaną (objętą procesem sterowania), która może pomóc w poszukiwaniu przyczyn takiego stanu. Osiągnięto znaczną poprawę jakości i to nie przez zmianę projektu produktu, czy zmianę ustawienia parametrów procesu, ale przez proste rozpoznanie możliwości sterowania w danych warunkach. Autor artykułu podkreśla, że sieć PCNN realizuje z 93% stopniem dokładności całkowitą diagnostykę procesu produkcyjnego. Nie odnotowano fałszywych alarmów.

Z coraz większym powodzeniem wykorzystywane są również sieci neuronowe w systemach eksperckich z logiką rozmytą, mające zastosowanie do sterowania w czasie rzeczywistym. Hung [13] proponuje zastosowanie sieci SOM i LVQ do zbudowania dwupoziomowo trenowanej sieci, która generuje reguły rozmyte na podstawie eksperymentalnych danych wejścia-wyjścia. Sieć ta była wykorzystywana do sterowania dynamicznym systemem odwróconego wahadła.

LITERATURA

[1] Beale M., Demuth H.: Preparing Controllers for Nonlinear Systems. AI Expert, July 1992
[2] Brause R.: Neuronale Netze. B.G. Teubner Stuttgart 1991
[3] Burgin G.: Using Cerebellar Arithmetic Computers. AI Expert, June 1992
[4] Caudill M.: The View From Now. AI Expert, June 1992
[5] Caudill M.: Embedded Neural Networks. AI Expert, April 1991
[6] Caudill M.: A Little Knowledge is a Very Dangerous Thing. AI Expert, June 1993
[7] Caudill M.: Avoiding the Great Backpropagation Trap. AI Expert, July 1991
[8] Caudill M.: Putting Time in a Bottle. AI Expert, February 1993
[9] Caudill M.: Driving Solo. AI Expert, September 1991
[10] Freeman, Skapura: Neural Networks, Addison Wesley Publishing Company
[11] Hinton G.: Jak sieci neuropodobne uczą się na podstawie doświadczeń. Świat Nauki, listopad 1992
[12] Hiotis A.: Inside a Self Organizing Map. AI Expert, April 1993
[13] Hung Chuan-Chang: Building a NeuroFuzzy Learning Control System. AI Expert, November 1993
[14] Neural Network Resource Guide. AI Expert, February 1993
[15] Plummer J.: Tighter Process Control with Neural Networks. AI Expert, October 1993
[16] Rock D., Malkoff D., Stewart R.: AI and Aircraraft Health Monitoring. AI Expert, February 1993
[17] Tadeusiewicz R.: Sieci neuronowe. Akademicka Oficyna Wydawicza, Warszawa 1993.

INFORMATYKI nie ma w sprzedaży kioskowej, istnieje tylko w prenumeracie. Można ją zaprenumerować na wszystkich pocztach w kraju.

Jednak każdy numer czasopisma można kupić w Redakcji. Prowadzimy sprzedaż odręczną i wysyłkową. Wystarczy do nas zadzwonić (Warszawa tel. 39-14-34).

Dla spóźnionych rezerwujemy do sprzedaży egzemplarze począwszy od numeru styczniowego br.

Na życzenie odbiorcy wystawiamy fakturę z podatkiem VAT.

Wskazówki dla Autorów

Nadsyłane artykuły nie mogą być publikowane lub przeznaczone do opublikowania w innych czasopismach.

Materiał oprócz tekstu zasadniczego powinien zawierać:

* krótki życiorys zawodowy Autora i jego zdjęcie,
* wykaz literatury,
* tabele,
* materiał ilustracyjny (rysunki, zdjęcia czarno-białe, wydruki) dołączony do artykułu (nie wklejać materiału w tekst),
* podpisy pod ilustracje.

Na osobnej stronie prosimy podać: tytuł naukowy, nazwisko i imię, nazwę zakładu pracy, numer telefonu służbowego oraz informację, jaką drogą przekazać honorarium – kasa Wydawnictwa, poczta, konto.

W związku z naliczaniem podatku dochodowego prosimy także podać:

* oprócz nazwiska i imienia – także drugie imię,
* imiona rodziców,
* datę i miejce urodzenia,
* numer identyfikacyjny PESEL,
* miejsce zameldowania,
* adres Urzędu Skarbowego właściwego dla miejsca zamieszkania Autora.

Tekst artykułu prosimy dostarczyć w formie maszynopisu lub wydruku komputerowego, pisanego jednostronnie, z podwójnym odstępem (30 wierszy na stronie i 60 znaków w wierszu).

Rysunki powinny być czytelne i mogą być wykonane ołówkiem.

Firmy informatyczne w 1993 r.

dokończenie ze s. 25

Podsumowując można stwierdzić, że 1993 r. był kolejnym trudnym rokiem w rozwoju przemysłu informatycznego na świecie, który w sumie przyniósł jednak współczynniki rozwoju lepsze niż w innych dziedzinach gospodarki światowej, ale znacznie gorsze aniżeli miał je przed kilkoma laty, w okresie swego bujnego rozwoju. Dla poszczególnych firm sytuacja przedstawiała się różnie. Można mówić o pierwszych oznakach zahamowania regresu dużych firm oraz o bardziej dynamicznym rozwoju tych, które szybciej dostosowały się do nowych trendów.

Źródło: Users to Vendors: Quality, Quality, Quality. Datamation June 15, 1994, s. 38–64

JAN RYŻKO

Ogłoszenia prosimy zgłaszać:

pisemnie
Red. INFORMATYKI
Pl. Inwalidów 10, p. 104, 01-552 Warszawa
lub
Dział Reklamy i Marketingu
ul. Mazowiecka 12,00-950 Warszawa
skr. poczt. 1004

telefonicznie
nr 39-14-34 lub 27-43-66
telefaksem
nr 39-14-34
26-80-16

z zaznaczeniem – Red. INFORMATYKI

Kwaśnicka H., Markowska-Kaczmar U.: Zastosowanie sieci neuronowych w systemach czasu rzeczywistego
INFORMATYKA 1994, nr 11, s. 3
Rodzaje, sposoby realizacji i problemy wyboru architektury sieci neuronowych oraz przykłady ich zastosowań w systemach czasu rzeczywistego.

Kwaśnicka H., Markowska-Kaczmar U.: Neural networks application in real-time systems
INFORMATYKA 1994, No. 11, p. 3
Kinds, realization manner and architecture choice problems of neural networks, as well as examples of their application in real time systems.

Kwaśnicka H., Markowska-Kaczmar U.: Neuronnetzeanwendung in Echtzeitsystemen
INFORMATYKA 1994, Nr. 11, S. 3
Arten, Realisationsmethoden und Architekturauswahlprobleme von Neuronnetzen, sowie Beispiele ihrer Anwendungen in Echtzeitsystemen.

Januchta M., Królikowski Z.: POSTGRES – rozszerzalny system zarządzania bazą danych
INFORMATYKA 1994, nr 11, s. 7
Charakterystyka opracowanego na Uniwersytecie Kalifornijskim w Berkeley systemu zarządzania bazą danych, zapewniającego rozszerzenie możliwości funkcjonalnych w porównaniu do istniejących konwencjonalnych systemów tego typu.

Januchta M., Królikowski Z.: POSTGRES – an extensible database management system
INFORMATYKA 1994, No. 11, p. 7
Characteristics of POSTGRES, a database management system developed at University of California, Berkeley, which secures extension of its functional possibilities compared with simular conventional systems.

Januchta M., Królikowski Z.: POSTGRES – ein ausdehnbares Datenbankverwaltungssystem
INFORMATYKA 1994, Nr. 11, S. 7
Eine Charakteristik von POSTGRES, einem in University of California, Berkeley, erarbeiteten Datenbankverwaltungssystem, das die Erweiterung von funktionellen Möglichkeiten im Vergleich mit ähnlichen konventionellen Systemen sichert.

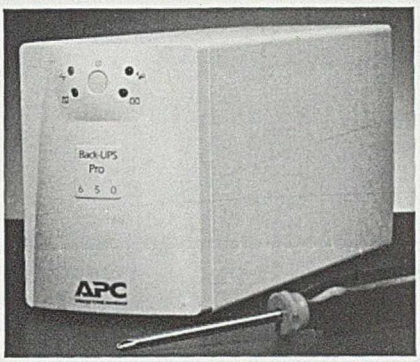
Kosmulska-Bochenek E.: Synchronizacja w rozproszonych systemach uwarunkowanych czasowo
INFORMATYKA 1994, nr 11, s. 17
Charakterystyka problemu oraz omówienie technicznych i organizacyjnych rozwiązań synchronizacji procesów w rozproszonych systemach uwarunkowanych czasowo.

Kosmulska-Bochenek E.: Synchronization in distributed real-time systems
INFORMATYKA 1994, nr 11, s. 17
Characteristics of the problem and discussion of contemporary technical and organization solutions of processes synchronization in distributed real-time systems.

Kosmulska-Bochenek E.: Synchronisation in zerstreuten Echtzeitsystemen
INFORMATYKA 1994, Nr. 11, S. 17
Eine Charakteristik des Problems und der heutigen technischen und organisatorischen Lösungen von Prozessesynchronisation in zerstreuten Echtzeitsystemen.

Back-UPS Pro- inteligentny zasilacz

Najnowszym produktem firmy American Power Conversion jest awaryjny zasilacz Back-UPS Pro. Urządzenie jest przeznaczone do ochrony takich urządzeń, jak wielozadaniowe stacje robocze, stanowiska inżynierskie CAD/CAM, stanowiska DTP, odpowiedzialne stacje robocze, kasy fiskalne, małe centralki telefoniczne, systemy multimedialne itp. Nowe modele łączą w sobie cechy znanych wcześniej UPS-ów serii Smart-UPS i Back-UPS. Jak informuje firma, seria Back-UPS Pro jest odpowiedzią na potrzeby krajów Europy Centralnej i Wschodniej, w których znana jest powszechnie zła jakość zasilania (nagłe spadki i skoki). Dodano w tej serii do Back-UPS przede wszystkim system Inteligentnego Zarządzania Baterią, sterowany przez wbudowany mikroprocesor. Informuje on m.in. o trybie pracy, przeciążeniu zasilacza, stanie baterii lub konieczności jej wymiany, wykrywa ewentualne usterki, automatycznie zabezpiecza przed zupełnym rozładowaniem baterii, skraca o połowę czas ładowania



baterii, umożliwia pracę bez używania baterii, gdy napięcie w sieci zasilającej zmienia się od 173 V do 296 V (AVR). Użytkownik może sam wymienić zużytą baterię na nową, co obniża koszty eksploatacji. Wszystkie modele Back-UPS Pro, z wyjątkiem najmniejszego, mają wbudowany interfejs pozwalający automatycznie i samodzielnie wyłączyć wielozadaniowy system operacyjny, gdy przerwa w zasilaniu przekroczy ustalony czas.

UPS-y zarządzane protokołami SNMP

Merlin Gerin, znany producent zasilaczy awaryjnych, wprowadził ostatnio trzy nowe produkty służące do zdalnego zarządzania zasilaniem w sieci przy użyciu protokołu SNMP. Program UM-View, pracujący w środowisku Windows, prezentuje w czasie rzeczywistym w formie ikon i wykresów dane o aktualnym stanie UPS-ów. Jego działanie jest uzależnione od modułu UM-Link, który jest agentem SNMP w sieci Ethernet. Program UM-Client umożliwia zdalne przeprowadzanie shutdown-u serwerów plików i dużych komputerów. Po zaniku zasilania z sieci energetycznej UM-Client otrzymuje informację o tym za pośrednictwem protokołu SNMP i przeprowadza automatyczne zamykanie plików systemowych w sposób określony uprzednio przez użytkownika. Służy też do planowanego uruchomienia systemu po powrocie zasilania. UM-Link, UM-View i UM-Client uzupełniają rodzinę zasilaczy firmy MG, Pulsar, Comet i Galaxy – korzystających z protokołu SNMP – o możliwości kontroli ważnych elementów sieciowych i usprawniają zarządzanie UPS-ami w sieciach LAN, WAN i MAN.

Firma Merlin Gerin jest pierwszym producentem UPS-ów, który uzyskał Certyfikat Zgodności IETF SNMP Interoperability Standard dla nowej rodziny inteligentnych produktów SNMP-UM-Link i UPS MIB.

Celebris z DEC-a

Digital Equipment Corporation wprowadza na rynek 5 nowych systemów PC z linii nazwanej Celebris. Budowane one są z myślą o aplikacjach wymagających szybkich komputerów, takich jak oprogramowanie do sporządzania raportów, analiz finansowych, księgowości, analiz i badań rynku oraz desktop publishing. Zastosowano w nich procesory od wersji i486 i Pentium, magistrale PCI i rozszerzone IDE, standardowo pamięć MB rozszerzalną do 128 MB, 256 KB zewnętrzne pamięci cache, rozszerzalne do 521 MB. Wyposażone są w 64-bitową kartę graficzną S3 z 1 MB wbudowanej pamięci, rozszerzalnej do 2 MB, oraz w mechanizm „wirtualnego okna" czyli oprogramowanie sterujące, które umożliwia równoległe otwarcie dwu pełnoekranowych okien.

„Harward Mark I" ma 50 lat

Mija 50 lat od narodzin pierwszego programowalnego kalkulatora IBM – Automatic Sequence Controlled Calculator, nazwanego „Harward Mark I". Jego twórcą był prof. Howard H. Aiken, który pracował nad nim od 1937 r. Kalkulator miał 15,5 m długości i 2,4 m wysokości. Zawierał ok. 800 km przewodów elektrycznych, 2 mln połączeń, 3,5 tys. wielostykowych przekaźników elektrycznych z 35 tys. styków, 1464 dziesięciobiegunowych przełączników i 72 sumatory. Łączył w swej konstrukcji urządzenia elektryczne i mechaniczne z opracowaną przez IBM techniką kart dziurkowanych. Mógł rozwiązywać problemy matematyczne, które dotychczas były uważane za nierozwiązywalne z powodu ogromnej zawiłości i konieczności przeprowadzania niesłychanie czasochłonnych obliczeń.

Osobiste systemy konferencyjne

Compaq Computer Corporation oraz PictureTel Corporation (firma założona w 1984 r. w Massachusetss, zajmująca się projektowaniem, produkcją i marketingiem produktów w zakresie telekomunikacji wizualnej) ogłosiły, iż podpisały porozumienie o pięcioletnim sojuszu strategicznym. Celem porozumienia ma być stworzenie i produkcja zgodnych ze standardem osobistych produktów konferencyjnych. Pierwsze produkty, które pojawią się na rynku światowym do 1995 r. będą odpowiadały standardowi telefonii ITU-TS1 (International Telecommunication Union – Telecommunications Standardization Sector) H.320 i będą mogły komunikować się z ponad 15 tysiącami stacjonarnych systemów wideokonferencyjnych, które już obecnie działają zgodnie z tym standardem. Compaq wniesie do tej spółki swoje doświadczenia na polu audio, grafiki i integracji systemowej, obecność w kanałach dystrybucyjnych oraz znaczny potencjał produkcyjny, PictureTel zaś swoją ekspercką wiedzę w zakresie opartych na standardach algorytmów kompresji, technologii audio klasy półprofesjonalnej oraz światowej integracji telefonicznej.

Upgrade do encyklopedii multimedialnej

Można już nabyć Microsoft Bookshelf 1994 – upgrade do multimedialnej biblioteki Microsoft na dysku CD-ROM (dostępne są wersje dla Microsoft Windows i Apple). Bookshelf '94 zawiera tekst siedmiu najnowszych wydań książek będących bestsellerami w kategorii źródeł informacji. Odtwarza m.in. nagraną głosem wymowę 80 tys. słów, oferuje ponad 60 animacji i wideoklipów przedstawiających zjawiska naukowe i wydarzenia historyczne. Narzędzia z grupy QuickShelf pozwalają użytkownikowi na uzyskanie dostępu do haseł Bookshelfu podczas pracy z dowolnym programem dla Windows lub Macintosha za pomocą jednego kliknięcia myszą. Wszystkie siedem publikacji jest zintegrowanych – użytkownik może poznać definicję i usłyszeć wymowę każdego ze słów zawartego w Bookshelf '94. Możliwe jest też kopiowanie definicji, cytatów i obrazów do edytorów, arkuszy kalkulacyjnych i programów prezentacyjnych uruchamianych w systemie Windows lub Macintosh.

Opracowała:
KRYSTYNA KARWICKA-RYCHLEWICZ

Międzynarodowe Targi Poznańskie

zapraszają na

INFOSYSTEM'95

Międzynarodowe Targi Elektroniki, Telekomunikacji i Techniki Komputerowej
10–13 kwietnia 1995

W 1994 r. w Targach wzięło udział 479 wystawców z 11 państw.
Zgłoszenia do 30 listopada – 25% rabatu

Międzynarodowe Targi Poznańskie
ul. Głogowska 14, 60-734 Poznań
tel.: 0-61 66-43-14, fax: 0-61 66-07-07, telex: 41 32 51 targ pl

479/94

PIERWSZY POLSKI INTELIGENTNY
SYSTEM INFORMATYCZNY WSPOMAGAJĄCY ZARZĄDZANIE
PRODUKCJĄ - (TPPiRP), FINANSAMI - KOSZTAMI, SPRZEDAŻĄ, KADRAMI
WRAZ Z SYSTEMEM INFORMACJI KIEROWNICTWA
WSPOMAGANY SYSTEMEM SZTUCZNEJ INTELIGENCJI PC-SHELL
ORAZ EKSPERCKĄ BAZĄ WIEDZY EKONOMICZNO-FINANSOWEJ - ISAF
(DLA DUŻYCH PRZEDSIĘBIORSTW PRODUKCYJNYCH)

Perfect-Ekspert AITECH

(DZIAŁAJĄCY W OPARCIU O ŚRODOWISKO PRACY - WYŁĄCZNIE SYSTEM OPERACYJNY DOS, NetWare v3.11, v4.01 FIRMY NOVELL)

OFERUJE

PRZEDSIĘBIORSTWO PRODUKCYJNO-USŁUGOWE INNOWACJI TECHNICZNYCH

JUNISOFTEX Sp. z o.o.

ROK ZAŁOŻENIA 1987
WARTOŚĆ SPRZEDAŻY wg F-02 1992 (14,6 mld), wg F-01 na 1993.10.31 (15,4 mld)
TEL/FAX 315-131, 315-700, 321-525, 388-052, 388-054 TLX 036233
BIURO TECHNICZNO-HANDLOWE
44-100 GLIWICE ul. Kościuszki 1
WSZYSTKICH ZAINTERESOWANYCH ZAPRASZAMY NA MERYTORYCZNE ROZMOWY DO NASZEGO BIURA TECHNICZNO-HANDLOWEGO
OFERUJEMY RÓWNIEŻ INTELIGENTNE NARZĘDZIE DO PRODUKCJI EKSPERCKICH BAZ WIEDZY

ARTIFICIAL INTELLIGENCE

PC-SHELL

DLA JEDNOSTEK BUDŻETOWYCH (NAUKA, OŚWIATA, URZĘDY MIAST I GMIN, SŁUŻBA ZDROWIA) BEZ PRAWA DO KOMERCYJNEGO WYKORZYSTYWANIA W CENIE 5.000.000,- (BEZ VAT) - INFORMACJE I ZAMÓWIENIA PROSIMY SKŁADAĆ BEZPOŚREDNIO U PRODUCENTA ARTIFICIAL INTELLIGENCE LABORATORY - AITECH, KATOWICE 14 SKR. POCZT. 2804.
DLA POZOSTAŁYCH JEDNOSTEK GOSPODARCZYCH BEZ PRAWA KOMERCYJNEGO WYKORZYSTYWANIA TJ. WYŁĄCZNIE NA WŁASNE POTRZEBY DO KOŃCA I KW. 1994 ROKU W CENIE PROMOCYJNEJ 10.000.000,- (BEZ VAT), DLA KUPUJĄCYCH DO CELÓW KOMERCYJNYCH DO KOŃCA I KW. 1994 25.000.000,- (BEZ VAT), LICENCJA OGRANICZONA NA NIEOKREŚLONĄ ILOŚĆ ODBIORCÓW KOŃCOWYCH.
POSZUKUJEMY DO WSPÓŁPRACY LUB NAWIĄŻEMY WSPÓŁPRACĘ Z EKSPERTAMI Z DOWOLNYCH DZIEDZIN NAUKI I TECHNIKI.

0/3/93